Actifio CLI Cookbook

Published October 12, 2018

Contents

Preface

This guide provides step-by-step instructions on how to use the Command Line Interface (CLI) to add, edit and delete Actifio appliances, virtual management servers, application groups and so on. It assumes you have access to the ***Actifio CLI Reference***, which is available on any Actifio appliance and in the ActifioNOW user portal.

Actifio Appliances

Unless otherwise specified, all features and functions described in this document apply to all Actifio appliances.

The ActifioNOW Customer Portal

During the configuration and initialization of your Actifio appliance your Actifio representative provided you with a user name and password for the ActifioNOW customer portal.

From the ActifioNOW customer portal you can obtain detailed reports about your Actifio appliance, access the Actifio product documentation, including release notes, and search the knowledge base for answers to specific questions.

To log into the ActifioNOW customer portal:

1. Go to: **https://now.actifio.com**.
2. When prompted, enter the user name and password provided by your Actifio representative.

Actifio Support Centers

To contact an Actifio support representative, you can:

- Send email to: support@actifio.com
- Call:
  From anywhere: +1.315.261.7501
  US Toll-Free: +1.855.392.6810
  Australia: 0011 800-16165656
  Germany: 00 800-16165656
  New Zealand: 00 800-16165656
  UK: 0 800-0155019

Conventions used in this document

"Quoted Text"	Quotes from people, quoted computer output.
command	Commands entered at the command line interface.
option	Options to a command. As in "the -h option to the **udstask mkhost** command".
argument	Arguments to the option. As in "-type generic to the **udstask mkhost** command".
terminal output	Output from commands executed and displayed on the terminal window.

1 Introduction to the Command Line Interface (CLI)

This chapter includes command line interface commands such as **udsinfo** and **udstask**.

Nearly every action conducted in the Actifio Desktop can be translated directly into **udsinfo** and **udstask** commands. There are some commands that can only be done from the command line. For example, setting a custom expiration date for a single image out of a range of images pertaining to an application can only be done from the CLI.

For a list of commands using **udsinfo**, you can type **udsinfo -h** at the CLI. Each command also has a help section which can be accessed using `udsinfo COMMAND -h`

For a list of commands using **udstask**, you can type **udstask -h** at the CLI. Each command also has a help section which can be accessed using `udstask COMMAND -h`

The udsinfo Command

The **udsinfo** command allows you to read or collect data from the Actifio environment. It allows you to check settings about configuration aspects of Actifio as well as other items that you would see in the Actifio Desktop.

Additionally, most **udsinfo** commands will allow a switch **-filtervalue** which will allow you to apply filters to the results.

The **udsinfo** command is informational only. You can run it without any danger to the system. It's a great way to learn more about the CLI.

Output from udsinfo

The **udsinfo** command is used to collect information from an Actifio appliance. The output by default is tab delimited. While tab delimited output is easier to read for humans, computers are bit more finicky so may choose to reformat this output. You can format this output using the **-delim** keyword and specifying the delimiter of your choice.

Examples

- `udsinfo lsbackup -delim ,`
- `udsinfo lsuser -delim +`

Filtering Output from udsinfo

In some cases, you may only want to see a subset of the data you're requesting. Here, you can use **-filtervalue** to select exactly what you're looking for.

Examples

- `udsinfo lsbackup -filtervalue 'expiration>2018-10-31'`
- `udsinfo lsuser -filtervalue name=mike.w*`

In the first example, we retrieve the list of backups where the expiration date is after October 31, 2018. Here we use single quotes '' to provide the '>' as a literal. Excluding the single quotes would redirect the output to a file named "2015-10-31" which would not be the intent. I could have also executed the command this way:

udsinfo lsbackup -filtervalue expiration\>2015-10-31

which would escape the '>' and act as the comparison operator "greater than".

In the second example, we retrieve the list of users and filter on the name of the user. In this case, we're looking for any users with "mike." in their name. This would provide results such as:

- mike.waltham
- mike.warren

The **-filtervalue** is case sensitive so a user like "Mike.West" will not appear in the results.

> ***Note:*** *Filter values do not allow you to choose the columns displayed.*

We will see more examples of **-filtervalue** later in this text.

The udstask Command

The **udstask** command allows you to conduct actions in the Actifio environment. This allows you to modify existing settings or create new settings as well as other actions that you would conduct in the Actifio Desktop.

Since **udstask** commands are actions, there is no **-filtervalue** option.

In this section we will explore:

- Listing Users
- Listing Information for a User
- Changing User Details
- Adding a Generic Host

Listing Users

After logging into the CLI, to view all the user accounts on an Actifio appliance, you can use the following command:

udsinfo lsuser

The output of this command is a tab delimited list of user information.

```
    id clienabled lastname firstname timezone         externalid email name       isprotected
comments dataaccesslevel password

    1 true     Admin   System   America/New_York            admin    true      admin
0 ******

 27661 false                  America/New_York            act_clu5-b false
0 ******

 83686 false                  America/New_York             spadmin   false
0 ******

161371 false    User    QA      America/New_York             qauser    false
0 ******
```

To view a comma separated output, use the **-delim** keyword as follows:

udsinfo lsuser -delim ,

```
id,clienabled,lastname,firstname,timezone,externalid,email,name,isprotected,comments,dataacces
slevel,password

1,true,Admin,System,America/New_York,,,admin,true,admin,0,******

27661,false,,,America/New_York,,,act_clu5-b,false,,0,******
```

```
83686,false,,,America/New_York,,,spadmin,false,,0,******
161371,false,User,QA,America/New_York,,,qauser,false,,0,******
192758,false,,,America/New_York,,,snoop.Bhat,false,,0,******
```

To view a comma separated ouput and filter out users whose name contains 'admin' use **lsuser** as follows:

```
udsinfo lsuser -delim , -filtervalue name=*admin*
id,clienabled,lastname,firstname,timezone,externalid,email,name,isprotected,comments,dataaccesslevel,password
1,true,Admin,System,America/New_York,,,admin,true,admin,0,******
83686,false,,,America/New_York,,,spadmin,false,,0,******
```

Listing Information for a User

In the previous examples, we looked at all users or a subset of users within Actifio. Each example provided output in a tab delimited or comma delimited format. Since we know the 'id' value from the output above, we can use it in the command to view details of a particular user.

Use the id of the 'admin' user to get more details as follows:

```
udsinfo lsuser 1
clienabled true
lastname Admin
firstname System
id 1
timezone America/New_York
externalid
email
name admin
isprotected true
comments admin
dataaccesslevel 0
password ******
```

> ***Note:*** *You can apply this convention to any output from a command where an argument is not presented.*

To list all the images on an appliance:

```
udsinfo lsbackup
```

To list all the details of an image with id = '12345':

```
udsinfo lsbackup 12345
```

Changing User Details

Use the **udstask chuser** command to modify the first and last name for the "admin" user.

```
udstask chuser -firstname IT -lastname Administrators admin
```

Use the **udsinfo lsuser** command to verify if the changes are done:

```
udsinfo lsuser admin
clienabled true
lastname Administrators
firstname IT
id 1
timezone America/New_York
externalid
email
name admin
isprotected true
comments admin
dataaccesslevel 0
password ******
```

Adding a Generic Host

In this example, we will add a simple Linux host. A host type must be specified when adding a host. Actifio supports the following types of hosts:

- generic
- hmc
- hpux
- hyperv
- isilon
- netapp
- openvms
- tpgs
- vcenter

We will use **udstask mkhost** to add the host with "generic" as the host type.

```
udstask mkhost -type generic -ipaddress 172.25.101.177 -hostname snoop-cent6-2
242375
```

The **mkhost** command will create a host with the given details. The output of the **mkuser** command is the id of the created host. You can verify this using the following command:

```
udsinfo lshost 242375
svcname
ostype Linux
dataip
ipaddress 172.25.101.177
uniquename snoop-cent6-2_242374_01401
osversion #1 SMP Wed Jul 15 10:13:09 UTC 2015
alternateip
```

```
id 242375
originalhostid 0
timezone GMT-0500
description
isclusterhost false
isproxyhost false
friendlypath snoop-cent6-2
vcenterhostid
modifydate 2015-10-30 11:30:43.700
isesxhost false
isvm false
hostname snoop-cent6-2
vmtype
properties 0
osrelease 2.6.32-504.30.3.el6.x86_64
hasagent true
isvcenterhost false
sourcecluster 590021138994
hosttype
maxjobs 0
type
connector.port 5106
connector.username
connector.password
connectorversion 6.1.4.56564 HotFix 829
installdate 2015-08-20 22:00:46
uploadversion
updatedate
modifydate
upgradestatus Upgrade Success
errormessage
errorcode 0
componentname
componentversion
```

Congratulations! You just added a new host. At this point, Actifio only knows some details about the host. If a Connector is installed, Actifio will probe and get the information about that Connector version.

Actifio communicates with hosts to protect Applications. In the next section, we will look at discovering applications on a given host.

2 Discovering Applications

This chapter includes information on discovering applications using the CLI.

Application Discovery is a way to inform Actifio about "Applications" that exist on a host. This host can be a virtual machine or a physical host where the Actifio Connector is already installed. Applications can be filesystems, databases, or network volumes on a given host. To Actifio, everything is an Application.

Discovery of applications is based on the following prerequisites:

1. Actifio Connector is installed on the host and is listening on port 5106.
2. Actifio Connector options include "Change Tracking Driver" for SQL Server hosts, Hyper-V hosts (including SCVMM), and Exchange mailbox hosts. This is a "full install" for the Connector.
3. Appropriate network and local firewall rules are in place to allow Actifio to communicate with the host on port 5106.

Application discovery can be done via a host that's already discovered within Actifio. In the previous section, we added a host. Actifio can communicate with the host to discover which applications reside on the host and can be protected.

To discover the applications that reside on the newly added host, use the **appdiscovery** command.

```
udsinfo appdiscovery -host 242375
```

The discovery process does not have any output for successful discovery. Only errors are displayed on the terminal. The most common error occurs when an Actifio Connector is not installed on the host.

```
ACTERR-010022 connection to 172.25.101.177:5106 failed
```

3 Listing Applications

This chapter includes commands for listing applications and gathering additional information about an application.

Listing Applications

Once the application discovery has been performed, you can look at what applications have been discovered. In Actifio terminology, everything that resides on a host is an application. This includes filesystems, NFS mounts, databases (Oracle, SQL, Exchange), and virtual machines (Hyper-V, VMware).

Listing All Applications

Use the **lsapplication** command to list all applications on a CDS based on your user rights.

> ***Note:*** *An administrator can see all applications.*

```
udsinfo lsapplication -delim ,
```

```
id,auxinfo,protectable,appversion,morecredentials,volumes,username,hostid,lastfailover,description,appname,sourcecluster,originalappid,apptype,failoverstate,friendlytype,ignore,networkname,networkip,pathname,isclustered,appclass,sensitivity
```

```
333609,esx-r4-u19.services.actifio.com,FULLY,,,include:[Rack2-v3700-DS4] snoop-cent6-2/snoop-cent6-2.vmdk,,333607,,,snoop-cent6-2,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0
```

```
333611,esx-r4-u17.services.actifio.com,FULLY,,,,,333610,,Report Manager 6.1.1 Beta,snoop-jasper-01,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0
```

```
333614,esx-r4-u17.services.actifio.com,FULLY,,,,,333612,,,snoop-mysql-01,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0
```

```
333619,esx-r4-u17.services.actifio.com,FULLY,,,,,333615,,,snoop-ofiler,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0
```

```
333622,esx-r4-u18.services.actifio.com,FULLY,,,,,333620,,,snoop-ora-01,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0
```

```
333624,esx-r4-u16.services.actifio.com,FULLY,,,,,333623,,,snoop-ra602-2,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0
```

```
333626,esx-r4-u17.services.actifio.com,FULLY,,,,,333625,,Report Manager LA,snoop-ra612-ORIG,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0
```

```
333631,esx-r4-u19.services.actifio.com,FULLY,,,,,333630,,Automate tasks for VMware vSpher...,snoop-vco601,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0
```

```
333634,esx-r4-u17.services.actifio.com,FULLY,,,,,333632,,,snoop-w2k12-
1,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0

333636,esx-r4-u19.services.actifio.com,FULLY,,,,,333635,,,snoop-
w2k8r2,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0

333649,esx-r4-u17.services.actifio.com,FULLY,,,,,333637,,,snoop-w2k8r2-
2,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0

333652,esx-r4-u18.services.actifio.com,FULLY,,,,,333650,,,snoop-
w2k8sql,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0

338177,,FULLY,,,,,333650,,,C:\,590021138994,0,FileSystem,normal,FileSystem,false,,,,false,,0

338178,,FULLY,,,,,333650,,,TESTDB3,590021138994,0,SqlServerWriter,normal,SQLServer,false,,,WIN
2K8-SQL\SQLSERVER,false,SQLServer,0

338179,,FULLY,,,,,333650,,,TESTDB2,590021138994,0,SqlServerWriter,normal,SQLServer,false,,,WIN
2K8-SQL\SQLSERVER,false,SQLServer,0

338180,,FULLY,,,,,333650,,,TESTDB1,590021138994,0,SqlServerWriter,normal,SQLServer,false,,,WIN
2K8-SQL\SQLSERVER,false,SQLServer,0

338181,,FULLY,,,,,333650,,,AdventureWorks2008R2,590021138994,0,SqlServerWriter,normal,SQLServe
r,false,,,WIN2K8-SQL\SQLSERVER,false,SQLServer,0

338182,,FULLY,,,,,333650,,,ReportServer$SQLSERVERTempDB,590021138994,0,SqlServerWriter,normal,
SQLServer,false,,,WIN2K8-SQL\SQLSERVER,false,SQLServer,0

338183,,FULLY,,,,,333650,,,ReportServer$SQLSERVER,590021138994,0,SqlServerWriter,normal,SQLSer
ver,false,,,WIN2K8-SQL\SQLSERVER,false,SQLServer,0

338184,,FULLY,,,,,333650,,,msdb,590021138994,0,SqlServerWriter,normal,SQLServer,false,,,WIN2K8
-SQL\SQLSERVER,false,SQLServer,0

338185,,FULLY,,,,,333650,,,model,590021138994,0,SqlServerWriter,normal,SQLServer,false,,,WIN2K
8-SQL\SQLSERVER,false,SQLServer,0

338186,,FULLY,,,,,333650,,,master,590021138994,0,SqlServerWriter,normal,SQLServer,false,,,WIN2
K8-SQL\SQLSERVER,false,SQLServer,0

404707,,FULLY,,,,,333620,,,/,590021138994,0,FileSystem,normal,FileSystem,false,,,,false,,0

427589,esx-r4-u17.services.actifio.com,FULLY,,,,,427585,,,snoop-delete-
me,590021138994,0,VMBackup,normal,VMBackup,false,,,,false,,0

428297,,FULLY,,,,,333620,,,/boot,590021138994,0,FileSystem,normal,FileSystem,false,,,,false,,0

428298,,FULLY,,,,,333620,,,demodb,590021138994,0,Oracle,normal,Oracle,false,,,,false,Oracle,0

429290,,FULLY,,,,,333620,,,demodbB,590021138994,0,Oracle,normal,Oracle,false,,,,false,Oracle,0

429535,,FULLY,,,,,333650,,,E:\,590021138994,0,FileSystem,normal,FileSystem,false,,,,false,,0
```

The output is a comma delimited list of all the applications that the Actifio appliance is aware of.

> ***Note:*** *Use the* ***-delim*** *parameter to format the output for further analysis.*

Listing Applications Based on Type

To list the applications that are virtual machines, you can use the **lsapplication** command and filter on the "**apptype**" column with a value of "VMBackup".

```
udsinfo lsapplication -filtervalue apptype=VMBackup -delim ,
```

```
id,originalbackupid,appid,policyname,mountedhost,username,sourceimage,apptype,modifydate,jobcl
ass,flags,status,expiration,sourceuds,hostname,label,consistencydate,backupdate,backupname,slt
name,slpname,appname,prepdate,virtualsize,uniquehostname,componenttype,sensitivity

237292,0,112401,Dedup to Dedup 1,0,,,VMBackup,2015-10-27 14:40:49.748,dedup,36,succeeded,2015-
11-03 13:40:48.709,590021138986,shin-w2k8r2-2,,2015-10-27 14:36:14.000,2015-10-27
14:39:26.000,Clu5-D_0131328,Snap and DAR,Shin to snoop,Shin-W2K8R2-2,,80530636800,50093688-
3031-8cbd-048d-aaeeb897a4ed,0,0

239121,0,112401,Dedup to Dedup 1,0,,,VMBackup,2015-10-28 14:39:40.287,dedup,36,succeeded,2015-
11-04 13:39:39.301,590021138986,shin-w2k8r2-2,,2015-10-28 02:02:07.000,2015-10-28
14:39:34.000,Clu5-D_0132174,Snap and DAR,Shin to snoop,Shin-W2K8R2-2,,80530636800,50093688-
3031-8cbd-048d-aaeeb897a4ed,0,0

240824,0,112401,Dedup to Dedup 1,0,,,VMBackup,2015-10-29 14:39:40.891,dedup,36,succeeded,2015-
11-05 13:39:39.866,590021138986,shin-w2k8r2-2,,2015-10-29 02:02:04.000,2015-10-29
14:39:35.000,Clu5-D_0132701,Snap and DAR,Shin to snoop,Shin-W2K8R2-2,,80530636800,50093688-
3031-8cbd-048d-aaeeb897a4ed,0,0

243762,0,112401,Dedup to Dedup 1,0,,,VMBackup,2015-10-30 14:39:45.063,dedup,36,succeeded,2015-
11-06 13:39:44.114,590021138986,shin-w2k8r2-2,,2015-10-30 02:02:06.000,2015-10-30
14:39:39.000,Clu5-D_0133254,Snap and DAR,Shin to snoop,Shin-W2K8R2-2,,80530636800,50093688-
3031-8cbd-048d-aaeeb897a4ed,0,0

245400,0,112401,Dedup to Dedup 1,0,,,VMBackup,2015-10-31 14:42:42.276,dedup,36,succeeded,2015-
11-07 13:42:41.018,590021138986,shin-w2k8r2-2,,2015-10-31 02:03:38.000,2015-10-31
14:39:43.000,Clu5-D_0133790,Snap and DAR,Shin to snoop,Shin-W2K8R2-2,,80530636800,50093688-
3031-8cbd-048d-aaeeb897a4ed,0,0

247022,0,112401,Dedup to Dedup 1,0,,,VMBackup,2015-11-01 13:43:37.213,dedup,36,succeeded,2015-
11-08 13:43:36.212,590021138986,shin-w2k8r2-2,,2015-11-01 02:02:12.000,2015-11-01
13:39:44.000,Clu5-D_0134363,Snap and DAR,Shin to snoop,Shin-W2K8R2-2,,80530636800,50093688-
3031-8cbd-048d-aaeeb897a4ed,0,0
```

This output only lists the virtual machines that Actifio is aware of.

Here are some other example commands for filtering on applications.

- List applications with "ora" in their name: `udsinfo lsapplication -filtervalue appname=*ora*`
- List applications on host 12345: `udsinfo lsapplication -filtervalue hostid=12345`
- List applications of type "Oracle" on a specific host:
  `udsinfo lsapplication -filtervalue 'appname=Oracle&hostid=12345'`

Note: *Use the -delim parameter to format the output as per your requirement.*

App Types

ConsistGrp	Exchange	FileSystem	LVM Volume
Microsoft Hyper-V VSS Writer	nas	Oracle	SqlInstance
SQLServer	SystemState	VMBackup	

Note: *New App Types are added from time to time, so this list may be incomplete.*

Gathering Information about an Application

By default, **udsinfo lsapplication** only provides some basic information about the application. Use the application ID to retrieve more information about a specific application.

```
udsinfo lsapplication 428298
appclass Oracle
networkip
protectable FULLY
appversion
morecredentials
volumes
networkname
id 428298
mountedhost
username
lastfailover
description
originatingjob
parentappid 0
apptype Oracle
ignore false
auxinfo
frommount false
sensitivity 0
isclustered false
depth 0
hostid 333620
backupname
sourcecluster 590021138994
appname demodb
originalappid 0
pathname
failoverstate normal
friendlytype Oracle
username
password
```

4 Grouping Applications

This chapter includes commands to group applications using Groups and Consistency Groups.

Groups

A group is a logical grouping of applications that can span multiple hosts. A group is used to simultaneously apply protection for a large number of applications. Every application in a "Group" has its own individual set of jobs.

For example, you might group all the VMs that belong to a development team and protect them using a single Policy Template and Resource Profile.

To create a group, use the following commands:

```
udstask mkgroup

udstask mkgroupmember
```

To create a group, just choose a group name and create the group.

The number returned will be the group ID. Remember this.

In the example below, I create a group called 'productionVMs' and it gets a group ID of 9094296

```
user@docside> udstask mkgroup  -name productionVMs
9094296
```

I confirm the group exists:

```
user@docside> udsinfo lsgroup -filtervalue id=9094296
     id modifydate                description name
9094296 2018-10-11 20:42:59.249               productionVMs
```

I have 3 production VMs I want to place in the group. I get their application IDs with 'udsinfo lsapplication.

They are appids 372241, 5032895 and 5032905.

I add them to the group. The number returned is the group member ID.

```
user@docside> udstask mkgroupmember -appid 372241 -groupid 9094296
9094315
user@docside> udstask mkgroupmember -appid 5032895 -groupid 9094296
9094317
user@docside> udstask mkgroupmember -appid 5032905 -groupid 9094296
9094322
```

I now confirm my work:

```
user@docside> udsinfo lsgroupmember -filtervalue groupid=9094296
     id modifydate               appid groupid
9094315                         372241 9094296
9094317                        5032895 9094296
9094322                        5032905 9094296
```

Consistency Groups

A consistency group is a logical grouping of applications that reside on a single host. A consistency group is used to apply protection for a number of applications on the same host. Applications in a Consistency Group are all protected using a single job. For example: The SQL Server databases in an instance can be protected using a consistency group and can all be protected using a single job.

To create a consistency group, use the following commands:

```
udstask mkconsistgrp

udstask mkconsistgrpmember
```

Almost every application can reside in a consistency group, but there are some exceptions. A consistency group cannot contain:

- Applications from multiple hosts.
- Multiple virtual machines.
- Multiple Oracle databases from a single host.

Remember that a Consistency Group can only contain applications from a single host. Applications from multiple hosts cannot be protected simultaneously as there is no possible way to guarantee consistency between systems. Multiple virtual machines also cannot be snapped simultaneously. Similarly, there is no mechanism within Oracle to quiesce and backup multiple databases at once.

Once a Consistency Group is created, it receives an "appid" and can be referenced using that appid from that point forward and can be treated like any other application.

Creating a Consistency Group

To create a Consistency Group we need the host ID. To learn this use `udsinfo lshost`.

In this example the hostname is Oracle, so we use this command to learn the id of the host, which is 4731. (Extra information irrelevant to this exercise has been omitted.)

```
user@docside> udsinfo lshost oracle
vcenterhostid 4460
...
id 4731
...
originalhostid 0
```

Now we know the host ID, we can create a consistency group with this command. The Group is called OracleProdCG and it is a group of Apps on host ID 4731. The returned number is the consistency group ID, remember this number. In this example it is 9094386

```
user@docside> udstask mkconsistgrp -hostid 4731 -groupname OracleProdCG
9094386
user@docside>
```

Note if the group has spaces, then use single quotes like this:

```
udstask mkconsistgrp -hostid 4731 -groupname 'Oracle Prod CG'
```

We can confirm our group using the ID like this (extra information irrelevant to this exercise has been omitted):

```
user@docside> udsinfo lsconsistgrp 9094386
description
hostid 4731
...
apptype ConsistGrp
...
```

```
groupname OracleProdCG
```

Now we can add members to the group. Remember we can only add apps from the same hostid.

We could confirm the appids using this command, using the host ID as a filter:

```
udsinfo lsapplication -filtervalue hostid=4731
```

Once we know the application IDs, we can add them.

For each application ID we add we will get a member ID returned. In this example we add app IDs 667422 and 667428

```
user@docside> udstask mkconsistgrpmember -groupid 9094386 -appid 667422
9094455
user@docside> udstask mkconsistgrpmember -groupid 9094386 -appid 667428
9094459
```

We can then check we have added all members with this command:

```
user@docside> udsinfo lsconsistgrpmember -filtervalue groupid=9094386
     id modifydate               appid groupid
9094455                         667422 9094386
9094459                         667428 9094386
user@docside>
```

5 Protecting Applications

This chapter includes commands to protect applications, list policy templates and resource profiles, make SLAs and on demand protection.

Protecting Applications

To protect applications with Actifio, you need to know three things:

1. What do you want to protect?
2. When do you want to protect it?
3. Where do you want to store this protected data?

The three items above translate directly into Actifio terminology like this:

1. The application to be protected with Actifio.
2. The schedule to protect an application is specified using Actifio Policy Templates.
3. The location to store protected data is specified using Actifio Resource Profiles.

Note: *Policy Templates and Resource files appear in the Actifio SLA Architect screen of the GUI.*

Listing Policy Templates

Policy templates determine the schedule by which Actifio will protect an application. Policy templates also determine the schedule by which data is replicated to another location. They can become quite complex but designing them in the GUI helps to reduce that complexity.

Use the **lsslt** command to list Policy Templates.

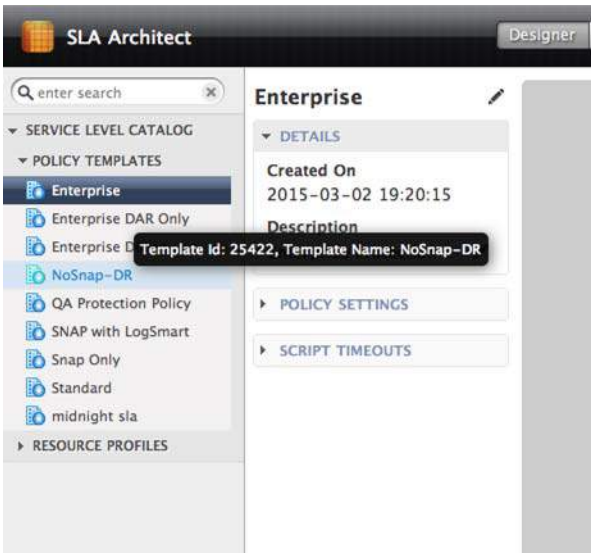
```
udsinfo lsslt
```

This shows us all the Policy Templates in the CLI.

```
    id override description           name
   103 true     Enterprise Protection Enterprise
   105 true     Standard Protection   Standard
 15260 true     Enterprise Protection Enterprise DAR Only
 25422 true     Enterprise Protection NoSnap-DR
 55320 true     Enterprise Protection Enterprise DBLog
```

```
155014 true     Enterprise Protection SNAP with LogSmart
162719 true     Enterprise Protection QA Protection Policy
167725 true     Enterprise Protection Snap Only
228426 true     Enterprise Protection midnight sla
```

In the Actifio Desktop, they are shown in SLA Architect. Here is an example showing the same information.



In some cases, you may want to create a policy that will never run. This is possible but a policy must be created for it. The idea behind a policy that never runs is to provide control to a scheduler outside the Actifio environment. This could be a simple cron job or something more complex like Tivoli Workload Scheduler, or Control-M.

Listing Resource Profiles

Resource Profiles specify a location (a disk pool in Actifio terms), where data will be stored when protected by Actifio. Resource Profiles also determine the location where the data will be replicated. In the examples below, we see three policies.

> ***Note:*** *The "remotenode" column indicates where the data is to be replicated.*

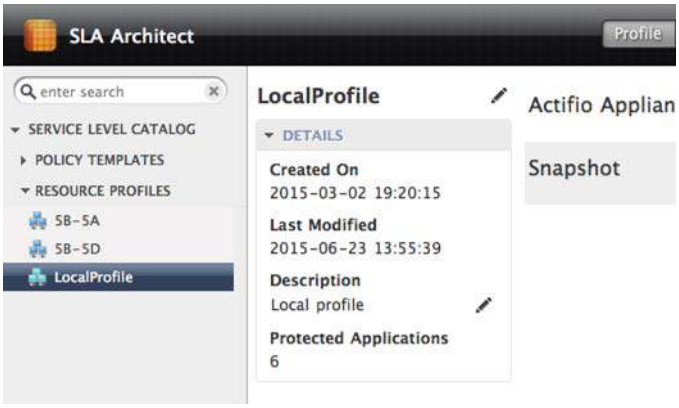
To list resource profiles, use:

```
udsinfo lsslp
```

This will list the Resource Profiles in the CLI.

```
    id description             name         performancepool primarystorage remotenode localnode
    51 Local profile           LocalProfile act_per_pool000                 none       Clu5-B
 18763 Local profile           5B-5A        act_per_pool000                 Clu5-A     Clu5-B
216041 New Profile Description 5B-5D        act_per_pool000                 Clu5-D     Clu5-B
```

In the Actifio Desktop, they are also shown in SLA Architect. Here is an example showing the same information.



Making SLAs

An SLA tells Actifio to protect the Application using the schedule from the lsslt output and the destination in **lsslp**. To make an SLA, we need to know the Application ID.

Let's take a simple application and protect it. For this example, we will use **mksla**. You need to know the Application ID, SLT ID, and SLP ID before protecting an application. You can retrieve the App ID via the GUI or via the CLI.

```
udsinfo lsapplication -filtervalue appname=/boot -delim ,
```

```
id,auxinfo,protectable,appversion,morecredentials,volumes,username,hostid,lastfailover,description,appname,sourcecluster,originalappid,apptype,failoverstate,friendlytype,ignore,networkname,networkip,pathname,isclustered,appclass,sensitivity
```

```
428297,,FULLY,,,,,333620,,,/boot,590021138994,0,FileSystem,normal,FileSystem,false,,,,false,,0
```

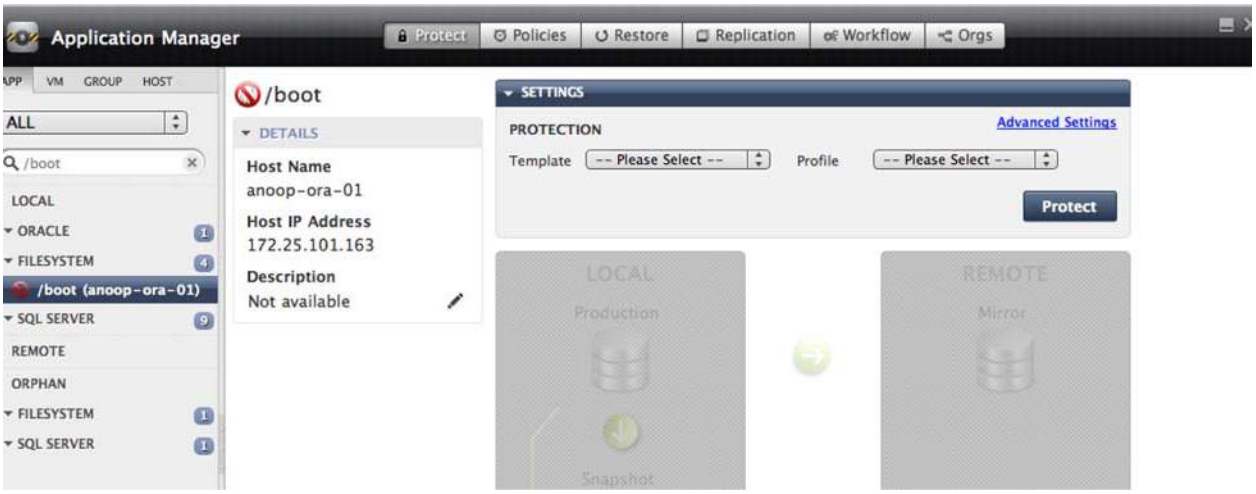
> ***Note:*** *The App ID in highlighted in bold in the output.*

Based on the schedules, I've chosen the "Enterprise" Policy Template and the "LocalProfile" Resource Profile for protecting this /boot application. Based on the output in Listing Policy Templates and Listing Resource Profiles, those IDs are:

SLT: 103

SLP: 51

So now I can set up protection for **/boot**. In the Actifio Desktop, this is what it looks like before protection.



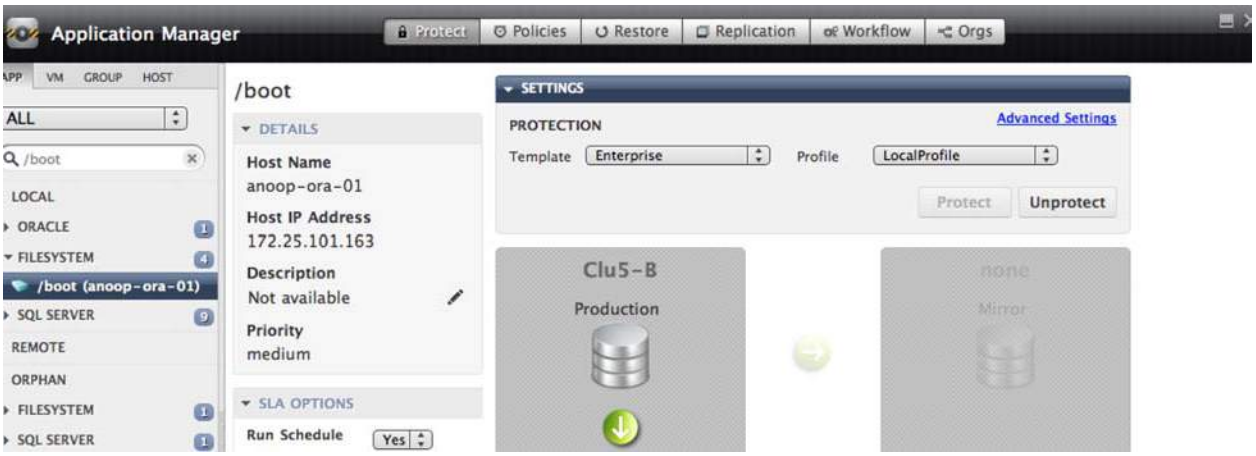
Note: *In the image above, the red No icon indicates that this application is not protected.*

Use the **mksla** command to protect the application.

```
udstask mksla -appid 428297 -sltid 103 -slpid 51
```

434597

The output is the ID of the SLA that's created. In the Actifio Desktop, the red No icon has been replaced with an icon of a filesystem.



Note: *The Template and Profile on the right side of the GUI no longer say "-- Please Select --" but show the actual SLT and SLP chosen in the* ***mksla*** *command above.*

Common error messages you may see when executing **mksla** are:

```
ACTERR-010016 object not found:
```

Indicating that the SLTID or SLPID may be incorrect.

```
ACTERR-010017 already protected
```

Indicating that the Application you're trying to protect may already have an SLA assigned.

```
ACTERR-010016 object does not exist:
```

Indicating that the App ID provided doesn't exist. Double check the App ID using the **lsapplication** command above.

Now that we've protected an application, the schedule set by the Policy Template (or SLT) will determine when a backup cycle will run. But what if we wanted to run a job on-demand? We'll explore that in the next section.

On Demand Protection

Let's look at how to run a backup on-demand for an application. This concept is slightly different from protecting an application as you must specifically call out a policy within a policy template that will be used for the backup. For that, we need to understand what policies are present in our Policy Template.

Recall that our protection in the previous section involved an SLT with ID 103. We will use **lspolicy** to see what policies are present in the SLT with ID 103.

```
udsinfo lspolicy -filtervalue sltid=103
```

```
id,endtime,rpo,scheduletype,description,priority,encrypt,name,retention,starttime,exclusion,ex
clusioninterval,policytype,repeatinterval,retentionm,selection,sltid,rpom,exclusiontype,op

131,23:55,24,daily,Snap daily, retain for 3 days,medium,,S-
Daily,3,02:00,,1,normal,1,days,,103,hours,none,snap

132,23:55,24,daily,Dedup daily, retain for one week,medium,,D-
Daily,1,02:00,,1,normal,1,weeks,,103,hours,none,dedup

133,23:55,24,weekly,Dedup weekly (Sat), retain for o...,medium,,D-
Weekly,1,02:00,,1,normal,1,months,daysofweek:sat,103,hours,none,dedup

134,23:55,24,monthly,Dedup monthly (1st), retain for ...,medium,,D-
Monthly,6,02:00,,1,normal,1,months,daysofmonth:1,103,hours,none,dedup
```

The SLT 103 has four policies with IDs 131, 132, 133, and 134. Notice the "op" column in bold and the values for each policy. They are "snap", "dedup", "dedup", and "dedup" respectively. This indicates that the "Enterprise" Policy Template (103) has one snap policy and three dedup policies.

In order to run a backup on-demand, we'll have to choose one of these policies. In most cases, you will want to run a snapshot policy rather than a dedup policy. Snap policies are responsible for capturing the data from the source itself. This could be a full backup if the application has never been backed up before or an incremental backup if the application has had a previous backup.

We do not need to know if an application has had a full or an incremental backup before. Actifio knows what state the application is in and will take the appropriate action. All we need to do is tell Actifio to run a backup of an application. We will run **backup** command with the correct policy and the app id.

```
udstask backup -app 428297 -policy 131
```

```
Job_0434845
```

This will immediately start a job in Actifio that will begin to take a snapshot of the /boot application. The output of the command is also the job name which can be used to reference a status.

> ***Note:*** *The policy that you select must be part of the SLT that is being used to protect the application.*

In this case, policy 131 (S-Daily) is a valid policy in the Enterprise Policy Template.

Operation Types

snap	dedup	directdedup
directonvault	verification (verify of a dedup image)	
cloud (OnVault)	replicate (Dedup-DR or Remote-Dedup)	
dedup_async	stream_snap	

6 Job Information

In the previous section, we ran an on-demand protection job using the **udstask backup** command. In this section, we'll check on the job that resulted and also look at historical job information.

Jobs in Progress

To view the jobs that are currently "running" or "canceling", we will use the **lsjob** command.

```
udsinfo lsjob -delim ,
```

```
id,virtualsize,progress,queuedate,jobname,expirationdate,appid,parentid,policyname,jobcount,de
scription,changerequest,priority,isscheduled,jobclass,flags,relativesize,status,hostname,pid,s
tartdate,retrycount,sltname,appname,sourceid,targethost,errorcode,enddate,jobtag,consistencyda
te
```

```
434845,100,6,2016-02-03 18:25:02.934,Job_0434845,2016-02-06 18:25:02.933,428297,0,S-
Daily,0,,IGNORE,medium,false,snapshot,0,100,running,snoop-ora-01,6283,2016-02-03
18:25:02.934,0,Enterprise,/boot,Image_0434836,Image_0434845,,0,,,2016-02-03 18:25:02.000
```

This command returns all running jobs for all applications that we currently have access to see. That is, if the currently logged in account is limited to only see applications in a certain organization, then we will only see the jobs that pertain to those applications.

In the example above, there is only one job running. That job has an id of "434836" with jobname as "Job_0434836".

To get additional details of a job, you can pass the ID to the **lsjob** command.

```
udsinfo lsjob 434836
virtualsize 100
progress 11
queuedate 2016-02-03 18:25:02.934
currentstep 0
jobname Job_0434845
expirationdate 2016-02-06 18:25:02.933
appid 428297
parentid 0
policyname S-Daily
originaljobclass snapshot
id 434845
jobcount 0
priority medium
```

```
changerequest IGNORE
description
isscheduled false
jobtag
jobclass snapshot
flags 68
relativesize 100
status running
hostname snoop-ora-01
pid 6283
consistencydate 2016-02-03 18:25:02.000
startdate 2016-02-03 18:25:02.934
retrycount 0
totalsteps 0
sltname Enterprise
appname /boot
sourcecluster 590021138994
sourceid Image_0434836,Image_0434845
targethost
errorcode 0
enddate
```

> ***Note:*** *The "status" field has a value of "running" to indicate that the job is in progress.*

You can also use the -**filtervalue** option to look at a list of jobs. Here are some examples.

List all jobs for the host 'snoop-ora-01'. This would show all the jobs including those of jobclass snapshot, dedup, expiration, and remote-dedups, etc.

udsinfo lsjob -filtervalue hostname=snoop-ora-01

List all jobs for the host 'snoop-ora-01' with a jobclass of 'snapshot' (excludes all other jobclasses from the output).

udsinfo lsjob -filtervaue 'jobclass=snapshot&hostname=snoop-ora-01'

Jobs that have Completed

Not all jobs will finish successfully but all jobs should complete. Any jobs that are not running are listed using the **lsjobhistory** command. This is mostly for historical purposes so you can see what jobs may have failed in the past. The output of **lsjobhistory** will contain all the fields from the **lsjob** command and will include some additional fields.

The **lsjobhistory** command will give you a large list of all jobs that are no longer in the "running" state.

udsinfo lsjobhistory

List out all the jobs within the last 24 hours that were successful. Here you could replace status with "failed" or "canceled" to see jobs that have failed or have canceled.

udsinfo lsjobhistory -filtervalue 'startdate since 24 hours&status=succeeded'

List out all jobs in the last 24 hours that have succeeded for a particular application ID.

udsinfo lsjobhistory -filtervalue 'startdate since 24 hours&status=succeeded&appid=428297'

List out all jobs in the last 24 hours that have succeeded for a particular application ID and were of jobclass snapshot.

udsinfo lsjobhistory -filtervalue 'startdate since 24 hours&status=succeeded&appid=428297&jobclass=snapshot'

All of the above commands can be delimited for easy parsing as well.

Listing Backup Images for an Application

Now that we've run a backup of an Application, we should look at the details of that backup image. But first, let's see how we can list backups of all applications and of all types. For this, we will use **lsbackup**.

Listing all Images

To retrieve all backup images on an Actifio appliance execute the following command:

```
udsinfo lsbackup
```

This will give us all images that the Actifio appliance is aware of. The list will have the following jobclasses:

Snapshot: Backups of an application that are currently in the snapshot pool. This is the instant access pool.

Dedup: Backups of an application that are currently in the dedup pool of the appliance. This is the long term retention pool.

Remote-Dedup: Dedup backups of an application that have been replicated to another appliance's dedup pool.

DedupAsync: Backups of an application that are not only replicated to a remote appliance but also hydrated into that remote appliance's snapshot pool for instant access on the remote site.

Liveclone: An optional special type of image that's used for testing and development purposes. Usually in the snapshot pool or another pool of type snapshot.

Mount: Backups of an application that are currently being accessed via the snapshot pool.

Just like any other command, you can pick any image from the output of **lsbackup** and get more details about it.

Listing Individual Images

To retrieve details of a specific image, use the following command:

```
udsinfo lsbackup Image_0434845
appclass
appid 428297
backuplock 0
originatinguds 590021138994
policyname S-Daily
username onDemand
sourceimage
prepdate
mappedhost 0
componenttype 0
jobclass snapshot
status succeeded
expiration 2016-02-06 18:26:07.889
sensitivity 0
hostname snoop-ora-01
label
depth 0
uniquehostname 50098a17-7431-bf6b-3b0f-b7bd5417de00
backupname Image_0434845
slpname LocalProfile
virtualsize 125829120
restorelock 0
originalbackupid 0
id 434847
mountedhost 0
apptype FileSystem
```

```
modifiedbytes 0
modifydate 2016-02-03 18:26:19.525
flags 68
sourceuds 590021138994
expirytries 0
consistencydate 2016-02-03 18:25:52.000
backupdate 2016-02-03 18:25:02.000
targetuds 590021138994
sltname Enterprise
appname /boot
characteristic PRIMARY
transport SAN based, out-of-band storage
consistency-mode application-consistent
readyvm false
backuphost vq-ora-lin-a
```

Image Details:

```
nvolumes 1
logicalname /boot
restorableobject /boot
uniqueid dasvol:/boot
volumeUID 638A95F225800068C80000000000187A
target vdisk:fc-56B28C7E0F00
capacity 125829120
volumekey 0
isbootvmdk false
sourcemountpoint /boot
```

Filtering on Images

To filter the list of images use the -**filtervalue** keyword.

To list images in the snapshot pool, use the **lsbackup** command and specify the jobclass of snapshot.

udsinfo lsbackup -filtervalue jobclass=snapshot

The output includes all snapshots on an appliance.

Use this command to look for a specific application:

udsinfo lsbackup -filtervalue 'jobclass=snapshot&appid=428297'

This will give us those images that pertain to the "/boot" application that we protected in the previous section. We could have also used appname here but the result would be every single application named "/boot" not just the one we're looking for.

If you eliminate the "**jobclass**" filter, you would get all the jobs for that particular application.

To view all jobs for a particular host, use the command: **udsinfo lsbackup -filtervalue hostid=333620**

or **udsinfo lsbackup -filtervalue hostname=snoop-ora-01**

> ***Note:*** *Using the "hostid" guarantees a unique result as the "hostname" field need not always be unique.*

You can combine all the filters given above to further refine your search of the image list.

udsinfo lsbackup -filtervalue 'jobclass=snapshot&hostname=snoop-ora-01&appname=/boot'

> ***Note:*** *If the hostname and appname combination is not unique, then you must provide IDs instead of names.*

7 Mounting and Unmounting an Image

This chapter includes commands for mounting and unmounting an image.

Mounting an Image to a Host

In a previous section, we learned how to protect applications. Recovering the data is the next big challenge as recovery is the true test of whether the data was protected correctly or not.

Because most data is stored in the snapshot pool before going into the dedup pool, the data in the snapshot pool is accessible almost instantly. In this scenario, Actifio is not actually moving any data, which consumes time, it's leveraging the data that's stored natively in the snapshot pool and providing you with a virtual copy of a point in time for the selected application.

In Actifio terminology, this action of instantly accessing data is called a "mount". Let us conduct a mount of "**/boot**" from the previous section.Before we get started, let us view the filesystems on the host.

```
[root@snoop-cent6-2 ~]# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/vg_snoopcent62-lv_root
                      5.4G  1.1G  4.1G  21% /
tmpfs                 499M     0  499M   0% /dev/shm
/dev/sda1             477M   49M  404M  11% /boot
```

Note: *Notice the '/boot' application on the last line of the output.*

Now, let's "mount" the last image of /boot from the snapshot pool.

Use **udsinfo lsbackup**, to get the latest snapshot image name.

```
udsinfo lsbackup -filtervalue 'appid=428297&jobclass=snapshot'
```

From this, you can get the backupname value or id. In this example, it is "Image_0434845". Use the **udstask mountimage** command to mount that image.

To perform the mount, we need to know the host that we're going to mount to. In this case, we'll use the same host as the source.

```
udstask mountimage -image Image_0434845 -host snoop-ora-01
Job_0435884 to mount Image_0434845 completed
```

If we execute this in the CLI, the job starts immediately but we wouldn't know the Job ID or Job Name. This is because the shell is blocked until the job completes as this is a synchronous task. Any further instructions in your script wouldn't continue until the **mountimage** task is completed.

To complete the **mountimage** task in the background, you can use the -**nowait** switch.

```
udstask mountimage -image Image_0434845 -host snoop-ora-01 -nowait

Job_0435955 to mount Image_0434845 started
```

> ***Note:*** *The output with -nowait indicates "started" as opposed to "completed". It also gives the Job ID.*

If this job is started using -**nowait**, you should check on the job until it succeeds or fails before continuing. Once the job succeeds, if we go back to our terminal window, we should see a new mount appear

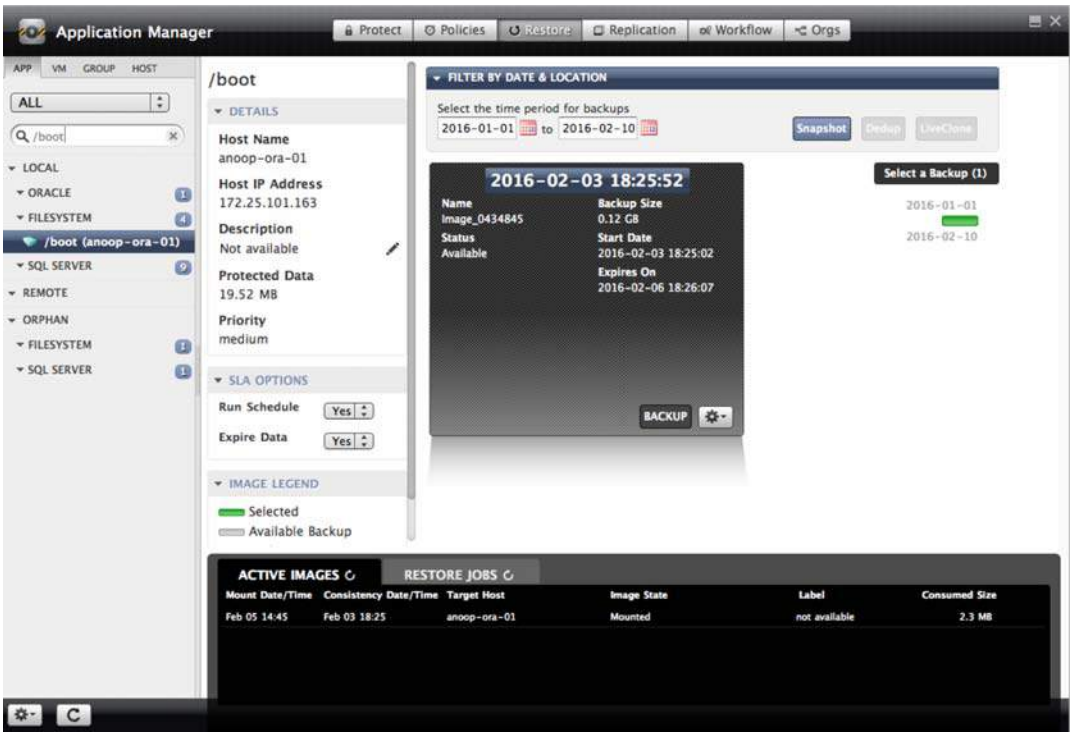
```
[root@vq-ora-lin-a ~]# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                       95G   21G   70G  23% /
/dev/sda1              99M   20M   75M  21% /boot
tmpfs                 5.9G  2.4M  5.9G   1% /dev/shm
/dev/sdb              117M   20M   91M  18% /act/mnt/Job_0435955_mountpoint_1454700944250
```

This new mount point represents a point in time snap of "/boot".

This example only deals with a filesystem and does not specify a target mount point. Complex examples can include Oracle or SQL databases that are mounted using App Aware mounts.

Viewing a Mounted Image

You can view a mounted backup image using the Application Manager from **Restore -> Active Images** for that specific application.



It is also visible from **Domain Manager -> Active Images -> Mounted/Unmounted**.

To view this data from the CLI you can use the "**lsbackup**" command to locate all Images on a given Appliance as well as remote appliances.

Use the lsbackup command and search for the mounted image using the "jobclass" filter with appid 428297.

```
udsinfo lsbackup -filtervalue 'jobclass=mount&appid=428297' -delim ,
```

```
id,originalbackupid,appid,policyname,mountedhost,username,sourceimage,apptype,modifydate,jobcl
ass,flags,status,expiration,sourceuds,hostname,label,consistencydate,backupdate,backupname,slt
name,slpname,appname,prepdate,virtualsize,uniquehostname,componenttype,sensitivity
```

```
435958,434847,428297,S-Daily,333620,onDemand,,FileSystem,2016-02-05
14:46:32.988,mount,68,succeeded,2100-01-01 00:00:00.000,590021138994,snoop-ora-01,,2016-02-03
18:25:52.000,2016-02-05 14:45:32.000,Image_0435955,Enterprise,LocalProfile,/
boot,,125829120,50098a17-7431-bf6b-3b0f-b7bd5417de00,0,0
```

Note: *The result returns one row with a matching 'appid' and jobclass ='mount'.*

To get more information about this "mount", you can use the **lsbackup** command with '435958' as the argument.

```
udsinfo lsbackup 435958
appclass
appid 428297
backuplock 0
originatinguds 590021138994
policyname S-Daily
username onDemand
sourceimage
prepdate
mappedhost 132733
componenttype 0
jobclass mount
status succeeded
expiration 2100-01-01 00:00:00.000
sensitivity 0
hostname snoop-ora-01
label
depth 0
uniquehostname 50098a17-7431-bf6b-3b0f-b7bd5417de00
backupname Image_0435955
slpname LocalProfile
virtualsize 125829120
restorelock 0
originalbackupid 434847
id 435958
mountedhost 333620
apptype FileSystem
modifiedbytes 0
modifydate 2016-02-05 14:46:32.988
flags 68
sourceuds 590021138994
expirytries 0
consistencydate 2016-02-03 18:25:52.000
backupdate 2016-02-05 14:45:32.000
```

```
targetuds 590021138994
sltname Enterprise
appname /boot
characteristic MOUNT
transport SAN based, out-of-band storage
consistency-mode application-consistent
readyvm false
mappedhostname esx-r4-u18.services.actifio.com
backuphost vq-ora-lin-a
```

Image Details:

```
nvolumes 1
logicalname /boot
restorableobject /boot
uniqueid dasvol:/boot
volumeUID 638A95F225800068C80000000000187F
target vdisk:rs-56B4FBD06300
capacity 125829120
volumekey 0
isbootvmdk false
sourcemountpoint /boot
mountedmountpoint /act/mnt/Job_0435955_mountpoint_1454700944250
mountedvdisk vdisk:rs-56B4FBD06300
```

Note: *The result provides the number of volumes as well as the mount point for each volume.*

Unmounting and Deleting a Mounted Image

In the previous section, we located our mounted image 435958. It was mounted back to "snoop-ora-01" as "/act/mnt/Job_0435955_mountpoint_1454700944250". When we choose to unmount this image, we have two options:

- Unmount
- Unmount & Delete

Unmount: Removes the mount point from the mount host, or target server, but stores that mount and any changes made to it in the Actifio storage pools for future use. The use case here is when you want to modify some data and then re-mount this image to another host or even the same host.

Unmount & Delete: Removes the mount point from the mount host, or target server, and also destroys any changes that were made to the mount. The use case here can be for running DBCC jobs on a database, validating the backups, copying a single file out of a backup.

To unmount a mounted image, we can use the "**unmountimage**" command.

```
udstask unmountimage -image 435958 -nowait
```

To unmount and delete a mounted image, we can use the same command with the option "-**delete**"

```
udstask unmountimage -image 435958 -delete -nowait
```

Because we specified "-nowait", the job runs in the background but we can track it because the job name is returned. If we had not specified "-nowait", the shell would be blocked until the command completed.

```
Job_0438204 to unmount Image_0435955 started
```

Now that we've looked at how to mount images, list mounted images, and unmount/delete mounted images, let's look at complex mount examples in the next section.

8 Application Aware Mounts

In a previous section, we discussed Mounts and how they work using a simple example of a filesystem being mounted back to a server. In this section, we will conduct an Application-Aware mount of an Oracle database and a SQL Server database.

Oracle Application-Aware Mounts

SQL Server App Aware Mounts

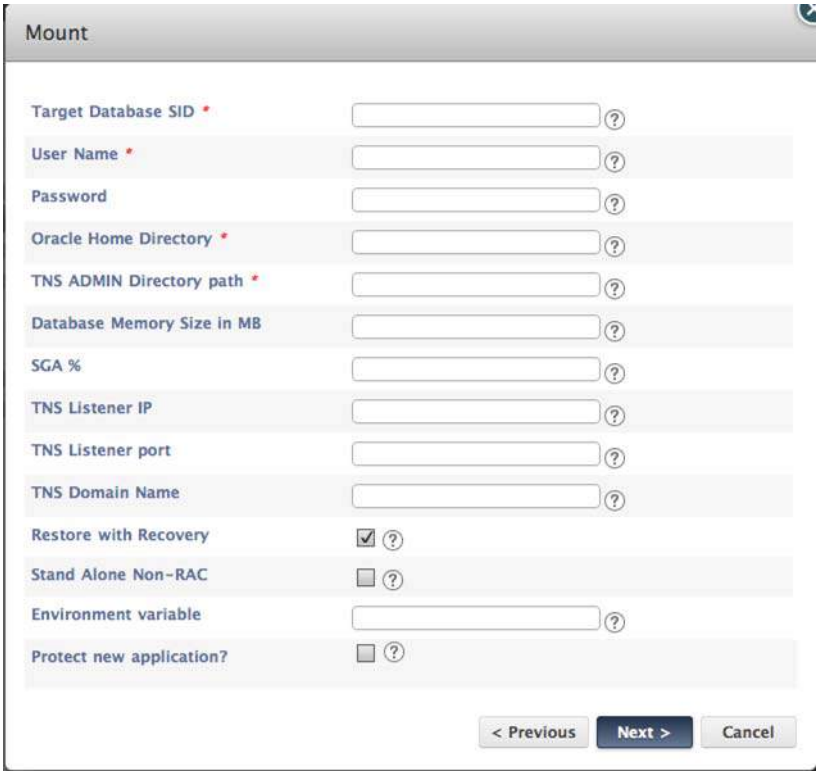
SQL Server Consistency Group App Aware Mounts

SQL Server Management Studio Databases

A simple mount provides access to the filesystems that Actifio has in its storage repository. It is mounted as a regular filesystem for browsing. App Aware mounts go one step further and do the work that would be manually done by the end user. Typically, that's creation of a database on the server and re-discovery of the application in Actifio so it can also be protected.

Oracle Application-Aware Mounts

We've already protected an Oracle DB called "demodb" for this exercise. Let's look at an example of an app aware mount of an Oracle database in the GUI.



When we click the App Aware Mount button, we are presented with some fields that may look familiar to an Oracle DBA. Filling out the required (*) fields result in a new DB created that's running off the Actifio presented disk.

To conduct an app aware mount from the CLI, use the **udsinfo lsappclass** command. Executing the "**udsinfo lsappclass**" command without any options provides us with a list of options we can specify.

```
udsinfo lsappclass -delim ,
name,friendly name,description
SQLServerGroup,SQL Server Group,Consistency group including SQL Server Databases and optionally one or more filesystems applications
SQLServer,SQL Server,SQL Server database
Oracle,Oracle,Oracle Database
OracleGroup,Oracle Group,Consistency group including one Oracle Database and optionally one or more filesystem applications
```

The name column is the option we can specify. In this case, we're interested in the 'Oracle' app class.

```
udsinfo lsappclass Oracle
name,type,label,description,required,group
databasesid,STRING,Target Database SID,SID for target database,true,
username,STRING,User Name,Oracle OS User name for database provisioning,true,login
password,ENCRYPT,Password,Password for Oracle OS user,false,login
orahome,STRING,Oracle Home Directory,Oracle Home Directory on target machine,true,
tnsadmindir,STRING,TNS ADMIN Directory path,TNS ADMIN Directory path (tnsnames.ora location path),true,
totalmemory,LONG,Database Memory Size in MB,Database total memory size in MB on target server,false,
sgapct,LONG,SGA %,Parameter to configure SGA/PGA memory when set,false,
tnsip,STRING,TNS Listener IP,TNS Listener IP: SCAN, VIP, or Host IP ,false,
tnsport,LONG,TNS Listener port,TNS Listener port (default 1521),false,
tnsdomain,STRING,TNS Domain Name,TNS Listener Domain name,false,
rrecovery,BOOLEAN,Restore with Recovery,Recover database after AppAware mount,false,
standalone,BOOLEAN,Stand Alone Non-RAC,Clone a stand alone non_RAC instance,false,
envvar,STRING,Environment variable,Environment variable can be separated by common delimiter,false,envar
```

Use this information to conduct the mount by referencing the help section of "`udstask mountimage -h`" and notice the "restoreoption" help text.

-restoreoption

(Optional) A comma delimited list of restore options where each restore option is a name=value pair. For available restore options use '**udsinfo lsrestoreoptions**'.

To perform an app-aware mount to a new application, additional options can be provided through an XML content, which needs to adhere to the rules of the appclass of the application. In addition, the app-aware mounted new application can also be optionally protected, with a new, or existing SLT and SLP, if so desired. The existence of the **provisioningoptions** indicates that this is an app-aware mount, regardless of -**appaware** flag. For example:

```
-restoreoption "provisioningoptions=<provisioningoptions>
    <databasesid>foodb1</databasesid>
    <orahome>/u01/app/oracle/product/11.2.0/db_1</orahome>
    <utlfiledirectory>/home/oracle</utlfiledirectory>
    <username>oracle</username>
  </provisioningoptions>,reprotect=true".
```

The mount command for an Oracle DB with app aware options is:

```
udstask mountimage -image 437991 -physicalrdm -host 333620 -nowait -restoreoption
“provisioningoptions=<provisioning-options>\
<databasesid>demodbA</databasesid>\
<username>oracle</username>\
<orahome>/home/oracle/app/oracle/product/11.1.0/db_1</orahome>\
<tnsadmindir>/home/oracle/app/oracle/product/11.1.0/db_1/network/admin</tnsadmindir>\
<rrecovery>true</rrecovery>\
</provisioning-options>,reprotect=false”
```

In the command the necessary fields for the App Aware mount to succeed are given below:

- databasesid - demodbA
- username - oracle
- orahome - /home/oracle/app/oracle/product/11.1.0/db_1
- tnsadmindir - /home/oracle/app/oracle/product/11.1.0/db_1/network/admin
- rrecovery - true

The last parameter instructed the command to open the database and reset logs.

Prior to running the **mountimage** command, the system process table had the following values:

```
# ps -ef | grep pmon
oracle    9907     1  0 Feb03 ?        00:00:01 ora_pmon_demodb
root     28948 28924  0 22:56 pts/1    00:00:00 grep pmon
```

Note: *There is only one pmon process for the source database “demodb”.*

After the job completes, a new ‘pmon’ process appears in the process table.

```
# ps -ef | grep pmon
oracle    9907     1  0 Feb03 ?        00:00:01 ora_pmon_demodb
oracle   29729     1  0 22:59 ?        00:00:00 ora_pmon_demodbA
root     29807 28924  0 22:59 pts/1    00:00:00 grep pmon
```

And just like the previous mount we conducted, a new filesystem also appears.

```
# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                       95G   21G   70G  24% /
/dev/sda1              99M   20M   75M  21% /boot
tmpfs                 5.9G  2.4M  5.9G   1% /dev/shm
/dev/sdb               50G  7.0G   40G  15% /act/mnt/Job_0439401_mountpoint_1455163155129
```

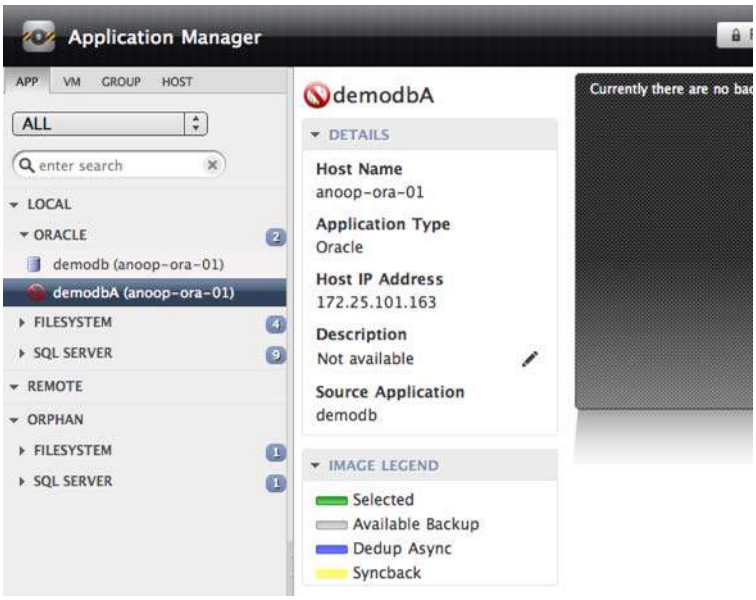
You can even log into this newly created database.

```
# su - oracle
$ export ORACLE_SID=demodbA
$ sqlplus / as sysdba

SQL*Plus: Release 11.1.0.7.0 - Production on Wed Feb 10 23:01:43 2016
Copyright (c) 1982, 2008, Oracle.  All rights reserved.
Connected to:
```

```
Oracle Database 11g Enterprise Edition Release 11.1.0.7.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> select name,open_mode from v$database;
NAME  OPEN_MODE
--------- ----------
DEMODBA   READ WRITE
SQL> select name from v$datafile;
NAME
--------------------------------------------------------------------------------
/act/mnt/Job_0439401_mountpoint_1455163155129/datafile/data_D-DEMODB_I-380433237
3_TS-SYSTEM_FNO-1_6pqtimts

/act/mnt/Job_0439401_mountpoint_1455163155129/datafile/data_D-DEMODB_I-380433237
3_TS-SYSAUX_FNO-2_6oqtimtd

/act/mnt/Job_0439401_mountpoint_1455163155129/datafile/data_D-DEMODB_I-380433237
3_TS-UNDOTBS1_FNO-3_6mqtimm5

/act/mnt/Job_0439401_mountpoint_1455163155129/datafile/data_D-DEMODB_I-380433237
3_TS-USERS_FNO-4_6rqtimud

NAME
--------------------------------------------------------------------------------
/act/mnt/Job_0439401_mountpoint_1455163155129/datafile/data_D-DEMODB_I-380433237
3_TS-DEMODB_TS_1_FNO-5_6nqtimsj
```

Note: *The datafiles listed are from the mount point listed in the "df -h" command above.*

In the Actifio Desktop, "demodbA" appears as a newly discovered and unprotected application.



Congratulations! We've just conducted our very first App Aware Mount of an Oracle Database on a filesystem. This is also possible with Oracle databases on ASM diskgroups and RAC instances with ASM. Combining an App Aware Mount with logs protection allows finer control over the point in time to recover to.

SQL Server App Aware Mounts

Just like Oracle databases in the previous section, SQL Server DBs can also have "App Aware" mounts. The logic is similar in both cases:

- Actifio presents a disk (or set of disks) that contains database files and information.
- Once the disk is accessible, the Connector finishes attaching the database or group of databases.

In this example, we'll conduct an App Aware mount of a single SQL Server database. The database to be mounted is "AdventureWorks2008R2" on host "snoop-w2k8sql".

Before we start, let's look at the fields required for an App Aware Mount via the GUI.

Observe that these fields are different from the Oracle App Aware Mount done earlier because they are platform specific and hence the term "App Aware".

To get the appid use the following command:

```
udsinfo lsapplication -filtervalue 'appname=AdventureWorks2008R2&hostname=snoop-w2k8sql' -delim
,
id,auxinfo,protectable,appversion,morecredentials,volumes,username,hostid,lastfailover,descrip
tion,appname,sourcecluster,originalappid,apptype,failoverstate,friendlytype,ignore,networkname
,networkip,pathname,isclustered,appclass,sensitivity
338181,,FULLY,,,,,333650,,,AdventureWorks2008R2,590021138994,0,SqlServerWriter,normal,SQLServe
r,false,,,WIN2K8-SQL\SQLSERVER,false,SQLServer,0
```

> ***Note:*** *The "appid" can be hard coded into scripts as once discovered it will not change.*

Use appid 338181 to get the latest snapshot.

```
udsinfo lsbackup -filtervalue 'jobclass=snapshot&appid=338181'
```

The latest snapshot is image id 439721. Now we need to know what options we can provide to "-restoreoption". For that, we'll go back to our appclass definitions to figure out what data is required.

```
udsinfo lsappclass SQLServer -delim ,
name,type,label,description,required,group
sqlinstance,EDITABLE_SELECT,SQL Server Instance Name,Name of target SQL Server Instance,true,
dbname,STRING,SQL Server Database Name,Name of target SQL Server Database,true,
```

```
recover,BOOLEAN,Recover Database After Restore,Bring database online after restore
operation,true,
username,STRING,User Name,User name for database provisioning,false,login
password,ENCRYPT,Password,Password for user,false,login
```

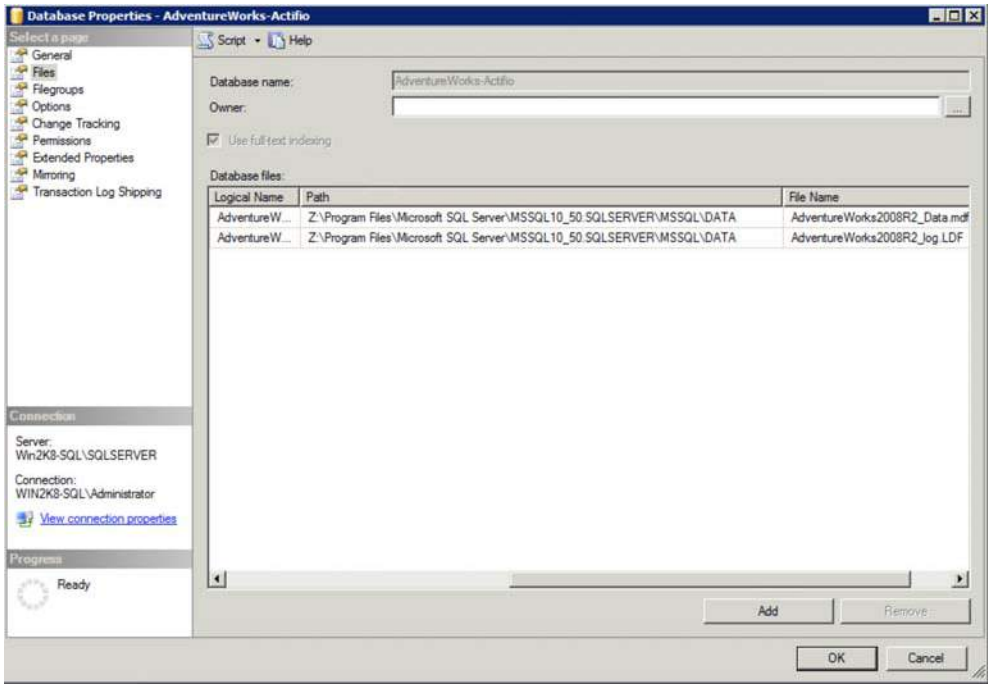
Use the **mountimage** command as follows:

```
udstask mountimage -image 439721 -host snoop-w2k8sql -physicalrdm -nowait -restoreoption
“provisioningoptions=<provisioning-options>\
<sqlinstance>WIN2K8-SQL\SQLSERVER</sqlinstance>\
<dbname>AdventureWorks-Actifio</dbname>\
<recover>true</recover>\
</provisioning-options>,reprotect=false”
```

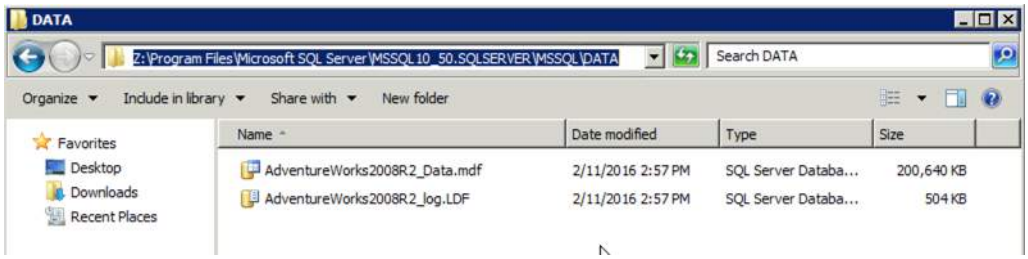
Once the job finishes, in SQL Server Management Studio, or via the command line:

```
PS C:\Users\Administrator> SQLCMD.EXE -S "localhost\SQLSERVER" -q "select name from
sys.databases"
name
--------------------------------------------------------------------------------------------------------------
master
tempdb
model
msdb
ReportServer$SQLSERVER
ReportServer$SQLSERVERTempDB
AdventureWorks2008R2
TESTDB1
TESTDB2
TESTDB3
AdventureWorks-Actifio
```

SQL Server Management Studio AdventureWorks-Actifio properties

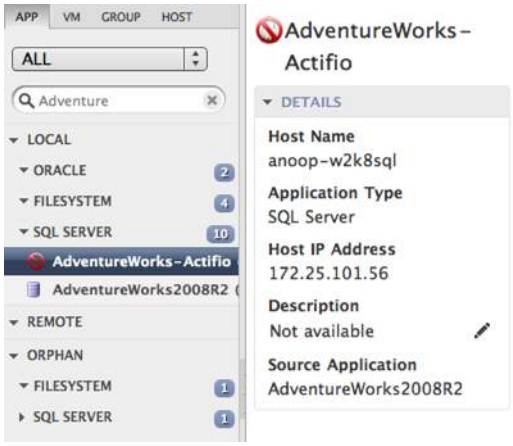


Note: *The files are on the Z:\ which you can easily browse in File Explorer.*



Congratulations! We just mounted our very first App Aware mount of a SQL Server database.

In the Actifio Desktop, you'll also notice a newly discovered, unprotected application called "AdventureWorks-Actifio".



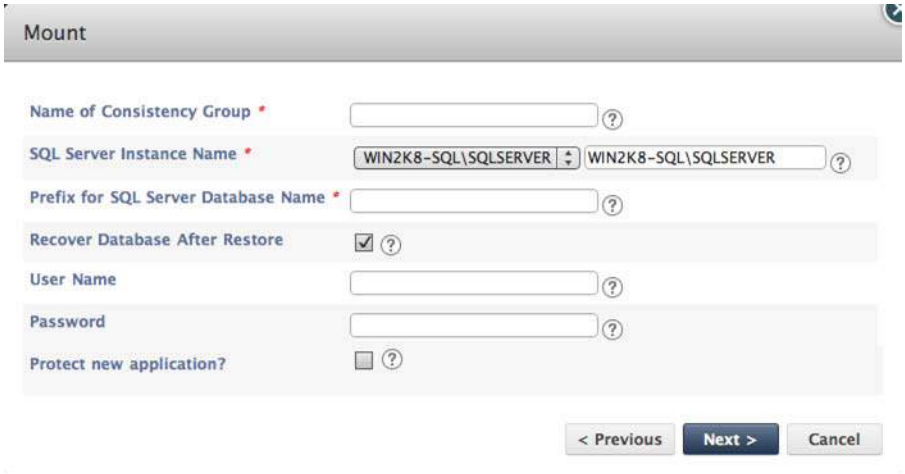
This example is for an app aware mount with a single node. App aware mounts with multiple nodes are also possible with the "**mountimage**" command.

In the next section, we'll tackle a variant of the SQL Server database. A group of them at the same time.

SQL Server Consistency Group App Aware Mounts

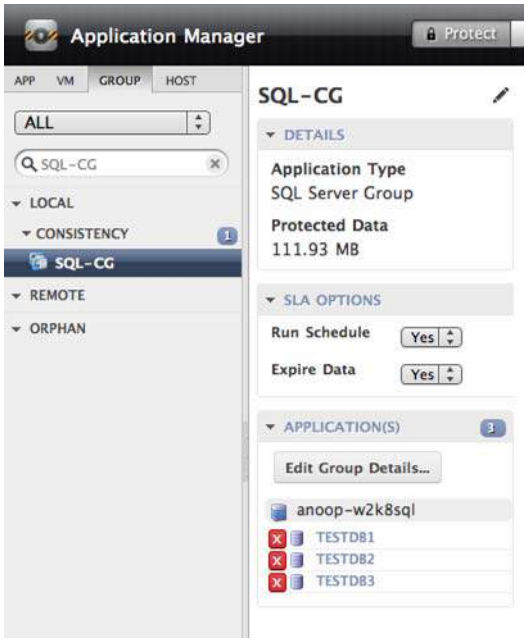
So we just finished our first App Aware Mount of a single SQL Server DB. What's next? How about a Consistency Group of databases? You may be asking why this is different from a single database App Aware mount. The answer is that it's only slightly different. We need to tell Actifio how to deal with naming the databases when they are mounted as well as the name of the resulting Consistency Group that is discovered in Actifio.

Let's look at the fields required in the GUI for SQL Server Consistency Group App Aware Mounts.



Compared to the mount of a single database, we'll have to specify some different fields.

We have a consistency group called "SQL-CG" with an "appid" of 338258. The details of this consistency group can be seen in the GUI.



From the command line, you can retrieve the members of this group with the following command:

```
udsinfo lsconsistgrpmember -filtervalue groupid=338258
    id modifydate               appid groupid
338261                         338180  338258
338263                         338179  338258
338265                         338178  338258
```

The output is a list of "appids" for this group.

Use the **lsbackup** command to get the image for this application.

```
udsinfo lsbackup -filtervalue 'appname=SQL-CG&jobclass=snapshot' -delim ,
```

```
id,originalbackupid,appid,policyname,mountedhost,username,sourceimage,apptype,modifydate,jobclass,flags,status,expiration,sourceuds,hostname,label,consistencydate,backupdate,backupname,sltname,slpname,appname,prepdate,virtualsize,uniquehostname,componenttype,sensitivity
440918,0,338258,S-Daily,0,onDemand,,ApplicationGroup,2016-02-11 18:48:44.854,snapshot,68,succeeded,2016-02-14 18:48:43.031,590021138994,snoop-w2k8sql,,2016-02-11 18:48:01.000,2016-02-11 18:43:00.000,Image_0440916,SNAP with LogSmart,LocalProfile,SQL-CG,,102944997376,50099c35-6ede-7bdc-106a-0a97681f8eca,0,0
```

Use the **lsappclass** command with "SQLServerGroup" to retrieve input values for the "**-restoreoption**" parameter.

```
udsinfo lsappclass SQLServerGroup -delim ,
name,type,label,description,required,group
ConsistencyGroupName,STRING,Name of Consistency Group,Name of Consistency Group,true,
sqlinstance,EDITABLE_SELECT,SQL Server Instance Name,Name of target SQL Server Instance,true,
dbnameprefix,STRING,Prefix for SQL Server Database Name,Prefix for target SQL Server Database Name,true,
recover,BOOLEAN,Recover Database After Restore,Bring database online after restore operation,true,
username,STRING,User Name,User name for database provisioning,false,login
password,ENCRYPT,Password,Password for user,false,login
```

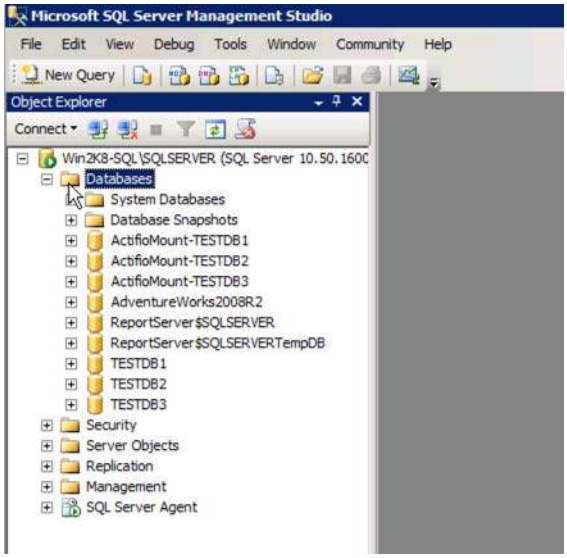
Use the **mountimage** command for the mount as follows:

```
udstask mountimage -image 440918 -physicalrdm -nowait -host snoop-w2k8sql -restoreoption "provisioningoptions=\
<provisioning-options>\
<ConsistencyGroupName>New-SQL-CG</ConsistencyGroupName>\
<sqlinstance>WIN2K8-SQL\SQLSERVER</sqlinstance>\
<dbnameprefix>ActifioMount-</dbnameprefix>\
<recover>true</recover>\
</provisioning-options>,reprotect=false"
Job_0441181 to mount Image_0440916 started
```

Once the job completes, you can observe the following in SQL Server Management Studio or using **sqlcmd**.

```
PS C:\Users\Administrator> SQLCMD.EXE -S $ENV:COMPUTERNAME\SQLSERVER -q "select name from sys.databases"
name
------------------------------------------------------------------------------------------------------------------------------------
master
tempdb
model
msdb
ReportServer$SQLSERVER
ReportServer$SQLSERVERTempDB
AdventureWorks2008R2
TESTDB1
TESTDB2
TESTDB3
ActifioMount-TESTDB1
ActifioMount-TESTDB2
ActifioMount-TESTDB3

(13 rows affected)
```

SQL Server Management Studio Databases



Note: *Our databases all have the same prefix, "ActifioMount-" before every database in this Consistency Group.*

In the Actifio Desktop, you will observe a newly discovered unprotected Consistency Group named "New-SQL-CG".