



ORACLE

Oracle ACFS

Advanced Cluster File System



Snapshot-based Replication How to Setup Guide

May, 2022 | Version 2.03
Copyright © 2022, Oracle and/or its affiliates
Public

DISCLAIMER

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

TABLE OF CONTENTS

Disclaimer	1
Introduction	3
Overview	4
Planning for ACFS Replication	4
Tagging Considerations	4
Choosing a Replication User	5
Step 1: Distributing a public key for repluser from primary to standby	5
Step 2: Getting the standby host_key	7
Step 3: Validating ssh-related key configuration	8
Step 4: Repeating key setup in the reverse direction	8
Using Replication Commands	10
Starting Replication	10
Pausing / Resuming Replication	13
Synchronizing the Primary and Standby Sites	13
Updating Replication Parameters	13
Administering Background Process	14
Comparing Mount Points	14
Failing Over Replication	15
Termination of Replication	16
Upgrading from Pre-12.2 to Snapshot-Based Replication	16
Appendix A – Case Study	17

INTRODUCTION

In Oracle Release 11.2.0.2 the ACFS File System Replication feature was introduced. ACFS replication enables replication of an ACFS file system across a network to a remote site. This capability is useful for providing disaster recovery capability. Similarly, to Data Guard, which replicates databases by capturing database redo, ACFS Replication captures ACFS file system changes on a primary file system and transmits these changes to a standby file system. Oracle ACFS replication functionality before release 12.2 replicated changes continuously, building on Oracle networking technologies, notably Network Foundation Technologies (NFT), to ensure connectivity between the primary and standby clusters.

This whitepaper describes how to configure Oracle ACFS snapshot-based replication (available since release 12.2 and above). Except where otherwise indicated, the whitepaper describes release 19.11 and later releases. The snapshot-based replication technology uses snapshots of the primary file system and transfers the differences between successive snapshots to the standby file system using the standard ssh command. This change in the design and implementation of Oracle ACFS replication introduces some differences in how replication is configured and used. First, the use of ssh means that some attention must be paid to setting up public host and user keys appropriately on the primary and standby nodes that will perform replication. Second, though we have tried to minimize changes to the replication CLI in designing the new technology, a few differences are still visible.

The combination of Oracle Real Application Clusters, Data Guard and ACFS Replication provides comprehensive site and Disaster Recovery policies for all files inside and outside the database.

OVERVIEW

Beginning in the 18c release, snapshot-based replication can process either an ACFS file system or a read-write snapshot of a filesystem. Either one of these is referred to as a storage location, or “*location*” for short.

In all releases, the source ACFS storage location is referred to as a primary and the target ACFS location as a standby. Replication works by taking snapshots of the primary. A first replication operation transfers the entire contents of the initial snapshot to the standby. Subsequent replication operations transfer just the delta between the most recent primary snapshot and the last previous snapshot.

Beginning in the 19.5 release, replication supports both switchover and failover. In these operations, the existing standby is converted to be the new primary, and the existing primary may either become the new standby, or may be replaced as the new standby by a different storage location.

As described below, replication requires user keys to be set up for ssh in both the primary and the standby clusters. In 19.10 and earlier releases, these keys needed to be set up for both the root user and the replication user (“*repluser*”). Starting in 19.11, these keys need to be set up only for “*repluser*”. This paper describes the situation as of 19.11.

In all cases, the standby file system is a read-only file system. One use-case for the standby file system is that it can be the source for backups.

PLANNING FOR ACFS REPLICATION

Examples described in this white paper assume that Grid Infrastructure software has been installed on nodes hosting the ACFS filesystem, and that the Oracle ASM Dynamic Volume Manager (Oracle ADVM) volumes and ACFS file systems are mounted. Note that a primary site running Linux, Solaris or AIX can replicate to a standby site running any of those operating systems. The primary and standby sites can have differing configurations; i.e., primary can be a multi-node cluster and standby can be a single-node cluster. If a standby node is used for disaster recovery purposes, then it is recommended that the standby node have a similar configuration such as cluster configuration.

There are no rigid primary or standby node roles; i.e. a primary node can provide the role of primary for one storage location and also provide the role of standby for another location. However, for simplicity, this paper will use the term “*primary node*” to indicate the node hosting the primary location, and “*standby node*” for the node hosting the standby location.

TAGGING CONSIDERATIONS

ACFS tagging is an important adjunct to ACFS replication. Rather than replicating an entire storage location, ACFS tagging enables a user to select specific files and directories for replication. Using tagging with ACFS replication requires that a replication tag be specified when replication is first initiated on the primary node. You can also add tags to files after replication has started. ACFS implements tagging with extended attributes. Note that some editing tools and backup utilities do not retain these extended attributes of the original file by default. Please review the ACFS Tagging section of the “*Oracle Automatic Storage Management Administrator’s Guide*” for the list of common utilities and their respective switch settings so that ACFS tag names are preserved on the original file.

Before implementing ACFS replication, it is important to determine how and what will be replicated; i.e., will all storage location data be replicated, only certain directories or only specific ACFS tagged files. This choice may affect file system sizing. ACFS tagging assigns a common naming attribute to a group of files. ACFS Replication uses this tag to filter files with unique tag names for remote file system replication. Tagging enables data or attribute based replication.

The following example illustrates recursively tagging all files of the “*/acfs*” directory with the tag “*reptag*”:

```
[root@primary-node1 ~]# /sbin/acfsutil tag set -r reptag /acfs
```

Keep in mind that the tags specified on the “*acfsutil repl init*” command line will be applied to files created after replication initialization, as well as to files that existed at the time of initialization. For example, you can replicate files with tags Boston and Milan, when at the time of replication only files with tag Boston exist; i.e., when no files exist with the tag Milan. Any subsequent files tagged with Milan will also begin to be replicated.

CHOOSING A REPLICATION USER

Oracle ACFS Replication starting in release 12.2 uses Secure Shell, also known as ssh, as the transport between the primary and standby clusters. Hence the user identity under which replication is performed on the standby needs to be carefully managed. A minimally-privileged user identity should be used. Generally, a user identity such as `sysasm` will already be defined on the standby node. That identity is suitable for use as the “replication user” – that is, the user that ssh will log in as on the standby node. If a different user is employed instead, that user should have “ASM admin privileges and Oracle install privileges”. Usually, this means that the user belongs to the same group as was specified to the installer when Oracle Grid Infrastructure (GI) was first installed. It should also be possible to log in as this user on the standby node. Here is an example of adding the user “repluser” to the existing group “asmadmin”:

```
# useradd -g asmadmin repluser
# passwd repluser
# mkdir -p /home/repluser/.ssh
# chown repluser:asmadmin /home/repluser/.ssh
```

In this article, we will use the name **repluser** to refer to the user that ssh will use to log in on the standby host. In using commands like the ones shown below, replace “*repluser*” with the actual user name that you chose to use in your deployment.

CONFIGURING PASSWORD-LESS SSH LOGIN

Why password-less ssh login

In the password-less ssh login method, we’re going to exchange encrypted keys instead of entering the actual password while connecting to the remote systems using ssh. So, nobody can easily hack or guess your password, because we don’t use passwords any more to access the remote systems. And more importantly, hidden key-loggers and brute-force attacks do not work if we use Password-less ssh.

Step 1: Distributing a public key for repluser from primary to standby

First, look for the file “`~repluser/.ssh/id_rsa.pub`” on your primary. This contains a public key for “*repluser*”. If the public key file exists, you should add it to the set of keys authorized to log in as “*repluser*” on each node of your standby using the `ssh-copy-id` command, as explained below. Append the key to the file “`~repluser/.ssh/authorized_keys`” on each standby node.

If the public key file does not exist, generate a public / private key pair on the primary by running this command as “*repluser*”:

```
[repluser@primary-nodel ~]$ ssh-keygen -t rsa

Generating public/private rsa key pair.
Enter file in which to save the key (~repluser/.ssh/id_rsa): ## Press Enter
Enter passphrase (empty for no passphrase): ## Enter Passphrase
Enter same passphrase again: ## Re-enter Passphrase
Your identification has been saved in ~repluser/.ssh/id_rsa.
Your public key has been saved in ~repluser/.ssh/id_rsa.pub.
The key fingerprint is:
b9:71:f7:e9:3f:8d:2a:34:67:fc:96:40:a1:2a:56:1c repluser@nodel
The key's randomart image is:
+--[ RSA 2048 ]-----+
|
|      E      . |
|      . . . . |
|      o . . . |
|      .S..o. |
|      o .+o.=. . |
|      . . . + ooo. |
|      . . . =.. |
|      ..o..o |
+-----+

```

The key files will be stored under your “*~repluser/.ssh*” directory. The above command will create two keys. One is private, and the other is public. The private key should be kept only in the directory where it was created -- you don’t have to transfer it to the remote standby systems. The public key should be transferred to all remote standby systems where replication may run. If the local primary system presents a public key that matches a key authorized to log in on the remote standby system, the primary will be able to access the standby. If the public key presented isn’t one that is authorized to log in, authentication will fail, and hence replication will fail too.

Now, copy the public key file to all your remote standby systems (using “*ssh-copy-id*” command):

```
[~repluser@primary-nodel ~]$ ssh-copy-id -i ~repluser/.ssh/id_rsa.pub repluser@<standby hostname>
```

Output (example):

```
$ ssh-copy-id -i ~repluser/.ssh/id_rsa.pub repluser@rwsam001
The authenticity of host 'standby (10.220.220.20)' can't be established.
RSA key fingerprint is 4c:95:a6:39:34:a1:ef:e7:d1:71:e9:05:b6:b6:e7:4a.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'standby,10.220.220.20' (RSA) to the list of known hosts.
repluser@standby's password:
Now try logging into the machine, with "ssh 'repluser@rwsam001'", and check in:

.ssh/authorized_keys
to make sure we haven't added extra keys that you weren't expecting.
```

If the “*ssh-copy-id*” command is not available, an alternative is to manually append the public key for “*repluser*” from each primary node to the “*~repluser/.ssh/authorized_keys*” file on each standby node.

Now, ssh to your remote standby as shown here. You should be able to access the remote system, getting the system date, without having to enter a password for “*repluser*”:

```
[repluser@primary-nodel ~]$ ssh repluser@<standby hostname> date
```

When your primary cluster has multiple nodes, you have the option of defining a separate key pair as described above for each node. Alternatively, you can define the key pair on one node, then copy the key pair to the other nodes. If you copy the key pair in this way, you must ensure that the public key on every node has permissions “0644”, and that the private key on every node has permissions “0600”. (Otherwise ssh will not use them.)

Step 2: Getting the standby host_key

A host key for each standby node where replication may run must be known for “repluser” on each primary node where replication may run. One way to generate the correct key is to run ssh manually as “repluser” from each primary node to each standby node. If the correct host key is not known already, then a warning displays and you can enable ssh to add the key. We saw this already when we ran ssh-copy-id in the previous steps. Running ssh-copy-id on a given primary node, targeting a given standby node, made a host key for that standby node known on that primary node. There's no need to do anything further to make a host key known.

Because the local node always connects to the remote node as user “repluser”, the host key for the standby node must be added to the “known_hosts” file for “repluser”.

The following is an example of obtaining a host key for “repluser”:

```
[repluser@primary-nodel ~]$ ssh repluser@standby date
The authenticity of host 'standby (10.137.13.85)' can't be established. RSA key fingerprint is
1b:a9:c6:68:47:b4:ec:7c:df:3a:f0:2a:6f:cf:a7:0a. Are you sure you want to continue connecting (yes/no)?
```

If you respond with yes, then the ssh setup is complete. A host key for host standby is stored in the known_hosts file (“~repluser/.ssh/known_hosts”) on the primary host for the user “repluser”.

After the host key setup for standby nodes is complete on a given primary node, you need to perform an additional step if you use a Virtual IP address (VIP) to communicate with your standby cluster. You must add the VIP name or address at the start of each line of the known_hosts file that refers to a host in the standby cluster. For example, if you use a VIP with the name standby12_vip, and your known_hosts file contains the following two lines that refer to your standby:

```
standby1,10.242.20.22 ssh-rsa
AAAAAB3NzaC1yc2EAAAADAQABAAQAC3pM2YTd4UUUEWEoCKDgGaTgsmPkQToDrdtU+JtVIq/96muvI
BaJUK83aqzeNIQkh+hUULsUdgKoKT5bxxrWYqhY6AlTEqNgBHjBrJt9C73BbQd9y48jSc2G+WQWyuI/
+s1Q+hIJdBNMxvMBQafisPWWUcaIx9Y/Jz1PgF61RP2cbfqAzixDot9fqRrAKL3G6A75A/6TbwmEW07d1zqQv
17ZGyeDYf5zQ72F/V0P9UgMEt/5DmcYTn3kTVGjOTbnRBe4A41Y4rVw5c+nZBDFre66XtORfQgWQB5ztW/Pi
08GYbcIszKoZx2HST9AZxYIAgcrnNYG2Ae0K6QLxxxScP
standby2,10.242.20.23 ssh-rsa
AAAAAB3NzaC1yc2EAAAADAQABAAQADIszcjzNtKN03SY8K1846skFTVP1HF/ykswbmkctEjL6KTWTW+NR
U4MGbvkbqgdXxuPCR7aogO2U3PEog1UVf3DWUoux8IRvqKU+dJcdTibMFkDAIhTnzb14gZ/lRTjn+GYsuP5
Qz2vgL/U0ki887mZCRjWVL1b5FNH8sXBUV2QcD7bjf98VXF6n4gd5UiIC3jv6l2nVTKDwtNHpUTS1dQAI+1D
tr0AieZTsuxXMaDdUZHGKDotjciMB3mCkKm/u3IFoioDqdZE4+vITX9G7DBN4CVPXawp+b5Kg8X9P+08Eehu
tM1BJ5lafy1bXoV1XUDLVIIFFBjNKRsqBvxxxxpS7
```

To enable the use of the VIP, you would modify these two lines to read as follows:

```
standby12_vip,standby1,10.242.20.22 ssh-rsa
AAAAAB3NzaC1yc2EAAAADAQABAAQAC3pM2YTd4UUUEWEoCKDgGaTgsmPkQToDrdtU+JtVIq/96muvI
BaJUK83aqzeNIQkh+hUULsUdgKoKT5bxxrWYqhY6AlTEqNgBHjBrJt9C73BbQd9y48jSc2G+WQWyuI/
+s1Q+hIJdBNMxvMBQafisPWWUcaIx9Y/Jz1PgF61RP2cbfqAzixDot9fqRrAKL3G6A75A/6TbwmEW07d1zqQv
17ZGyeDYf5zQ72F/V0P9UgMEt/5DmcYTn3kTVGjOTbnRBe4A41Y4rVw5c+nZBDFre66XtORfQgWQB5ztW/Pi
08GYbcIszKoZx2HST9AZxYIAgcrnNYG2Ae0K6QLxxxScP

standby12_vip,standby2,10.242.20.23 ssh-rsa
AAAAAB3NzaC1yc2EAAAADAQABAAQADIszcjzNtKN03SY8K1846skFTVP1HF/ykswbmkctEjL6KTWTW+NR
U4MGbvkbqgdXxuPCR7aogO2U3PEog1UVf3DWUoux8IRvqKU+dJcdTibMFkDAIhTnzb14gZ/lRTjn+GYsuP5
Qz2vgL/U0ki887mZCRjWVL1b5FNH8sXBUV2QcD7bjf98VXF6n4gd5UiIC3jv6l2nVTKDwtNHpUTS1dQAI+1D
tr0AieZTsuxXMaDdUZHGKDotjciMB3mCkKm/u3IFoioDqdZE4+vITX9G7DBN4CVPXawp+b5Kg8X9P+08Eehu
tM1BJ5lafy1bXoV1XUDLVIIFFBjNKRsqBvxxxxpS7
```

Ultimately, the host key configuration performed on this first node of your primary cluster must be performed on every node in your primary cluster; the result of the above sequence, or an equivalent, must exist on each primary node. One way to minimize the manual effort required to achieve this configuration is to update the “known_hosts” file on one node of the primary cluster, then copy the updated file to the other nodes of the cluster.

Note: By default, replication enables strict host key checking by ssh, to ensure that the primary node connects to the intended standby node or cluster when it runs ssh. However, if you are certain that this checking is unneeded, such as the case when the primary and standby clusters communicate over a private network, the use of strict host key checking by ssh

can be disabled. For information about disabling strict host key checking, refer to the “*sshStrictKey=no*” option of the “*acfsutil repl init primary*” command. If strict host key checking is disabled, then no host key setup is required.

Step 3: Validating ssh-related key configuration

After you have established the host and user keys for ssh, you can use the command “*acfsutil repl info -c -u*” to validate the keys. You run this command as “*repluser*”, on each node of the primary cluster. It takes as arguments all the hostnames or addresses on the standby cluster that the primary may use in the future to perform replication.

If you are not using a VIP to connect to your standby cluster, then for a given replication relationship, only one standby hostname or address is provided to “*acfsutil repl info -c -u*”. However, if future relationships may involve other standby host addresses, specify the complete set of addresses when running the “*acfsutil repl info -c -u*” command.

If you are using a VIP to connect to your standby cluster, then you should specify the names or host-specific addresses of all standby hosts on which the VIP may be active. Do not specify the VIP name or an address associated with the VIP. When replication uses ssh to connect to a VIP, the host key returned is the key associated with the host where the VIP is currently active. Only the hostnames or addresses of individual standby nodes are used by ssh in this situation.

The validation command has the following format. Here we show it being run as “*repluser*”:

```
[repluser@primary-nodel ~]$ acfsutil repl info -c -u repluser standby-addr1 [standby-addr2 ...] standby-mountpoint
```

The command confirms that “*repluser*” can use ssh to connect to each *standby-addr* given and log in as “*repluser*”, in the same way as replication does.

To validate the key setup for the *standby1* and *standby2* cluster presented in the previous command, you can use the following command:

```
[repluser@primary-nodel ~]$ acfsutil repl info -c -u repluser standby1 standby2 standby-mountpoint
```

The same command (specifying VIP hostname) would be used if you were planning to use the VIP “*standby12_vip*” to connect to the cluster.

If you plan to disable strict host key checking, you can skip this checking by adding the “*-o sshStrictKey=no*” option to the command line.

Step 4: Repeating key setup in the reverse direction

As noted earlier, ACFS replication supports switchover and failover, in which the primary and standby clusters reverse their “role” in performing replication. In order for these capabilities to work quickly and automatically when required, the ssh key setup steps 1 - 2 described above need to be performed in the opposite “direction”. That is:

- A public key for “*repluser*” must be distributed from standby to primary.
- A host key for each node of the primary must be known on each node of the standby.

Then step 3 should be performed, running “*acfsutil repl info -c -u*” on each node of the standby to verify that it can use password-less ssh to contact each node of the primary.

SSH Login Troubleshooting

NOTE ON PERMISSIONS FOR SSH-RELATED FILES

For ssh to work with the keys you have established, you must ensure that permissions are set properly on each node for the relevant “*~/ .ssh*” directory and some of the files the directory contains. On each primary node, as well as each standby

node, this refers to the “~/ .ssh” directory for “repluser”. In each case, the permissions for the files listed below must be at least as restrictive as shown in the `chmod` commands below:

```
$ chmod go-w $HOME
$ chmod go-rwx $HOME/.ssh
$ chmod go-rwx $HOME/.ssh/id_rsa
$ chmod go-rwx $HOME/.ssh/authorized_keys
```

Problem 1: In case the ‘ssh’ version is different between the primary and standby systems, you need to set permissions for ‘.ssh’ directory of your remote system. To do that, run the following command:

```
$ ssh repluser@<standby hostname> "chmod 700 ~/.ssh; chmod 640 ~/.ssh/authorized_keys"
```

Problem 2: If you still can’t ssh to remote systems for some reason, go to your standby remote system and enable ssh key authentication. To do that, on Linux, edit the following file: “/etc/ssh/sshd_config”. Find, uncomment and change the following lines as shown below:

```
PubkeyAuthentication          yes
AuthorizedKeyFile             .ssh/authorized_keys
ChallengeResponseAuthentication no
```

Save and close the file. Restart ssh service using command (on Oracle Linux 6):

```
[root@primary-node1 ~]# service sshd restart
```

(on Oracle Linux 7/8):

```
[root@primary-node1 ~]# systemctl restart sshd.service
```

NOTE ON SSHD CONFIGURATION

On some platforms, the ssh daemon “*sshd*” may be configured to log a message via “syslog” or a similar facility each time an ssh connection is established. To avoid this, the server configuration file “/etc/ssh/sshd_config” can be modified to specify a lower level of logging. The parameter that controls logging is called `LogLevel`. Connection messages are issued at level `INFO`. Any lower level (such as `ERROR`) will suppress these messages. So for instance adding this line to the file would suppress the messages: “`LogLevel ERROR`”

USING REPLICATION COMMANDS

The CLI for snapshot-based replication is similar to the one for pre-12.2 replication. There are differences in the interface to three commands: “*repl init*”, “*repl info*” and “*repl update*”. Also, note that, unlike in pre-12.2 replication, “*rep*” commands need not be run as *root*. Instead, these rules apply:

- The `acfsutil repl info` and `acfsutil repl bg info` commands may be run by any Oracle ASM administrator user.
- The `acfsutil repl compare` command is allowed to be run by any Oracle ASM administrator user, but should be run as *root* to maximize its access to the files being compared.
- All other `acfsutil repl` commands may be run either as *root* or as *repluser*.

Starting Replication

Once you are ready to initiate replication, the essential parts of the needed command lines are as follows. You will note that these command lines are similar but not identical to the equivalent command lines used in replication from before the 12.2 release.

In replication in 18c or later releases, a primary or a standby may be either a file system or a read/write snapshot of a file system. Either of these is called a storage location. You specify a file system using its mount point, as always. You specify a snapshot using the syntax “*snapname@mountpoint*”.

You initialize replication first in the standby cluster, and then in the primary cluster. On the standby, the only information you need to provide is:

- The name of the standby storage location
- The name of the replication user that will be used to replicate changes to the standby storage location

Most of the parameters affecting replication are specified on the primary. These include:

- The name of the primary storage location
- The ssh-style connection string (*username@network_addr*) to be used to connect to the standby cluster (“-s” option)
- The name of the standby storage location (“-m”)
- The “style” of replication, one of:
 - Interval-based, in which a replication operation will start once per specified interval (“-i”)
 - Constant-mode, in which a new replication operation starts as soon as the previous one ends (“-C”)
 - Manual-mode, in which replication occurs only when requested using “`acfsutil repl sync`” (“-M”)
- Optionally, the network endpoint to be used for the primary, when and if it ever serves as a standby, as following a failover operation – see below for more details (“-p”)

First, on the standby node, run a command like this one:

```
[joe@stdby-nodel ~]$ acfsutil repl init standby -u repluser /smntpt
```

This command specifies that the user *repluser* will be used to replicate changes to the file system mounted at “/smntpt”. The public key of *repluser*, as defined on the primary host, must be authorized for log in as user *repluser* on the standby.

Note: the acfs filesystem on standby must be empty at replication init time.

Next, on the primary node, run a command like this one:

```
[joe@primary-nodel ~]$ acfsutil repl init primary \  
    { -i interval | -C | -M } \  
    -s repluser@standby_host \  
    [ -m /smntpt ] /pmntpt
```

This command specifies that the user *repluser* will connect to the standby host in order to replicate the file system at primary mount point `/pmntpt` to the file system at standby mount point `/smntpt`. The interval at which replication sessions are started is specified with `-C` (constant mode) or `-i` (the interval between starting successive replication operations). When `-C` is given, a new replication session starts as soon as the previous session completes. Alternatively, `-M` may be given to say that replication sessions will start only on request.

When `-i` is given, the value interval has two parts – a count plus a unit indicator. The indicator must be one of `s`, `m`, `h`, `d` or `w` – for seconds, minutes, hours, days or weeks, respectively. So, for instance, `30s` indicates an interval of 30 seconds, `5m` indicates an interval of 5 minutes, and so on.

When the primary or standby cluster contains multiple nodes, and replication is to run on all nodes, it is recommended to use a VIP (like the SCAN VIP) instead of the address of a single host as the network endpoint specified to `acfsutil repl init primary`.

In the 19c release, by default, the network endpoint used for replication in the primary cluster will be the hostname of the node on which `acfsutil repl init primary` is run. To use a different endpoint, you must specify it with the option `-p`.

From the 21c release onward, by default, the SCAN VIP (if present) will be used as the network endpoint for replication in the primary cluster; if the SCAN VIP cannot be determined, the hostname of the node on which `acfsutil repl init primary` is run will be used. To use a different endpoint, you must specify it with the option `-p`.

A new option `-o param=value` option is used to override specific details of replication's operation.

The param strings currently defined are:

- **sshCmdPath** – the value string specifies the pathname at which the ssh command will be looked for.
- **sshCipher** – the value string specifies the cipher that will be passed to ssh to encrypt its sessions.

If tagging was enabled for this directory, then a tagname `reptag` can be added in the initialization command:

```
[joe@primary-node1 ~]$ acfsutil repl init primary \  
    { -i interval | -C } \  
    -s repluser@<standby hostname> \  
    [ -m /smntpt ] \  
    reptag \  
    /pmntpt
```

If the diskgroup compatible ADVM attribute is not at the required version, you may get an error such as:

```
acfsutil repl init: ACFS-03322: The ADVM compatibility attribute for the diskgroup is below  
the required version (12.2.0.0.0) for the 'acfsutil repl' commands
```

You could set up the required diskgroup compatible ADVM attribute as in the following (as grid user connected to ASM instance):

```
SQL> ALTER DISKGROUP DATA SET ATTRIBUTE 'compatible.asm' = '12.2.0.0.0', 'compatible.advm' =  
'12.2.0.0.0';
```

Or using ASMCMD:

```
$ asmcmd --nocp setattr -G DATA compatible.asm 12.2.0.0.0  
$ asmcmd --nocp setattr -G DATA compatible.advm 12.2.0.0.0
```

Verify that replication on the Primary File System has been initiated:

```
[joe@primary-nodel ~]$ /sbin/acfsutil repl info -c /pmntpt

Primary hostname:          primary-nodel
Primary path:              /pmntpt
Primary status:        Running
Background Resources: Active

Standby connect string:    repluser@standby
Standby path:              /smntpt

Replication interval:     0 days, 0 hours, 0 minutes, 0 seconds
Sending primary as of:    Fri Nov 06 05:33:22 2020
Status:                   Send Completed
Retries made:             0
Last send started at:     Fri Nov 06 05:33:22 2020
Last send completed at:   Fri Nov 06 05:33:22 2020
Elapsed time for last send: 0 days, 0 hours, 0 minutes, 0 seconds
Next send starts at:      now
Replicated tags:
Data transfer compression: Off
ssh strict host key checking: On
Debug log level:          3
```

Once the “acfsutil repl init” primary command completes successfully, replication will begin transferring copies of all specified files to the standby file system.

The average rate of data replication on the primary file system can be monitored using the “acfsutil repl info -s <mountpoint>” command, with the -s flag indicating sample rate. The amount of change includes all user and metadata modifications to the file system. The following example illustrates its usage:

The rate of data change on the primary file system can be monitored using the “acfsutil info fs -s <mountpoint>”

```
[joe@primary-nodel ~]$ /sbin/acfsutil repl info -s /pmntpt

-----
Fri Nov 06 05:33:22 2020 - Fri Nov 06 05:33:22 2020
-----

Data replicated:          0.00GB
Avg. rate of data replication: ~MB/minute
Avg. time from capture to apply: 00:00:00
```

Pausing / Resuming Replication

The `acfsutil repl pause` and `acfsutil repl resume` commands may be used only on the primary site. The `acfsutil repl pause` should be followed at a later time by `acfsutil repl resume`. The effect of the command `acfsutil repl pause` is to stop the operation of the replication daemon temporarily. Naturally, the effect of the command `acfsutil repl resume` is to resume the operation of the daemon.

If `acfsutil repl pause` is run while a replication operation is active, the operation will be allowed to complete – that is, the pause will not be effective until the current data stream has been applied on the standby. At that point, no more work will be performed for this replication until the `acfsutil repl resume` command is issued.

Synchronizing the Primary and Standby Sites

The replication command `acfsutil repl sync` can be used to synchronize the state of the primary site and the standby site. This command can only be run on the primary site. Users should first quiesce their applications and issue the OS `sync` command so that the synchronized state is known and meaningful to the user. The `acfsutil repl sync` command will then cause all outstanding replication data to be shipped from the primary site to the standby site. Specifically, the command will allow any in-progress replication operation to complete, and will then perform replication one final time to ensure that all changes on the primary have been replicated.

The command will return success when all of these changes have been successfully applied to the file system on the standby site. At this point, unless an unmount of the last node on the primary site cluster is about to occur (see next paragraph), applications may be restarted. The `apply` keyword is supported for backward compatibility, but has no effect on the operation of this command.

Successfully unmounting a replicated file system on the primary site does not imply that all file system changes made prior to the unmount have been successfully sent to the standby site or applied to the standby file system. If the primary file system is unmounted on one primary site node but remains mounted on one or more primary site nodes, changes to the file system made before the unmount continue to be transported to the standby site from the other nodes after the unmount. But if the administrator is unmounting the primary file system on the last primary site node where it is mounted and if he wishes to know that all file system changes made on the primary file system up to that point have been successfully applied to the standby file system, he may wish to quiesce applications on the primary site which modify the file system and then run `acfsutil repl sync`, to be sure that the standby file system is up to date. Then the unmount can be done.

Updating Replication Parameters

The `acfsutil repl update` command can be used to update all of the parameters established by the `acfsutil repl init` command, except for the mount point being used. In addition to the compression setting, these parameters include the replication interval, debugging level, and user or connect string used to communicate between primary and standby.

If run on the primary site, this command can be used to alter the username and network interface name (hostname or VIP name) used to connect to the standby site. It can also be used to change the replication interval or tracing level, change the cipher used by or the pathname used for ssh, and change whether compression is used.

In releases before 19.5, any new hostname or VIP name specified to connect to the standby site must resolve to the same cluster as the previous hostname or VIP name -- that is, this command may not be used to change to a different standby cluster.

If run on the standby site, this command can be used only to alter the username used by the primary to connect.

This command will return success when the updated information has been accepted on the local site. To change the username, the command should be run on both the primary and the standby sites. If the replication interval is changed using `-C` or `-i`, a replication operation will occur at the point when `acfsutil repl update` is run. Then the next replication operation will occur based on the newly-specified interval.

The `acfsutil repl update` command may be used for Oracle configuration changes. For example, the system administrator may need to change the network interface that is currently being used by replication to connect to the remote site. In this case, issuing the `acfsutil repl update` command will allow replication to continue running uninterrupted.

Administering Background Process

ACFS replication uses a background process on the primary cluster to transport file system changes to the standby file system. This process must be running for replication to function. The background process is started when replication is started via the “`acfsutil repl init`” command. The process is present and is registered with Oracle Clusterware whenever replication is active and the primary file system is mounted, so that the process will automatically be restarted after a reboot or system crash. This process is cluster wide.

To see if the ACFS replication background process is running at the primary site, issue the following command on the primary cluster

```
[joe@primary-nodel ~]$ acfsutil repl bg info /pmntpnt
Resource:          ora.repl.dupd.data.datclone.acfs
Target State:     ONLINE
Current State:    ONLINE on primary-nodel
```

In this example, the command reports that the background process is running. If it was not, the following message would be displayed:

```
[joe@primary-nodel ~]$ acfsutil repl bg info /pmntpnt
ACFS clusterwide replication background process is stopped
```

Comparing Mount Points

The “`acfsutil repl compare`” command verifies that all or part of the primary mount point has been replicated to the standby mount point. The standby mount point must be mounted locally to the primary (i.e. using NFS) for comparison. This command always checks all files on the primary file system against those on the standby file system.

If the “`-a`” option is used, the command also tests for extra files on the standby file system that do not exist on the primary. In this case, it will also try to check extended attributes. However, if NFS is used to mount the standby file system locally, the standby cannot be checked for matching extended attributes due to limitations in the NFS protocol.

The “`-t`” option is used when tags were specified during the “`acfsutil repl init`” operation. The “`-t`” operation locates all file names on the primary file system with the specified tag names and compares them to the corresponding files on the standby. If the “`-t`” option is used with the `all` argument, then all tag names supplied during the “`acfsutil repl init`” operation are selected. The “`-t`” option also tests for extra files on the standby file system that do not have an associated tag name that was specified during the “`acfsutil repl init`” operation. If NFS is used to mount the standby file system locally, the standby cannot be checked for matching tag names due to limitations in the NFS protocol.

The “`-a`” and “`-t`” options may not both be specified.

When “`-a`” or “`-t`” is given, the option “`-s`” may also be given to skip extended attribute comparisons for symlinks.

In all cases, the “`-v`” option may be given to cause the name of each file to be output after it is compared (verbose output).

Failing Over Replication

The `acfsutil repl failover` command is run in the standby cluster, and reverses the role of a replication standby location such that it becomes a replication primary location. The failover command insures that the standby location contains an exact copy of the results of the last successful replication transfer. If necessary, the command restores the location back to its state as of that transfer. `acfsutil repl failover` behaves differently based on the scenario in which it was run. We cover three scenarios:

1. Both the standby location and corresponding primary location are operating normally.

In this scenario, the command reverses the replication relationship. There is no data loss. Note that failover fails in this case if replication is paused. To enable this case to succeed, run `acfsutil repl resume`.

2. The primary location is not currently available but the user wants to wait until it is eventually is back online.

In this scenario, the command verifies the status of the replication primary. If the primary is not accessible and the timeout period has expired (if specified), then the command restores the standby location to its state as of the last successful replication transfer and converts it into a replication primary. Some data loss is possible, for example if there was a transfer in progress when the primary location became unavailable. When the original primary location becomes available, it is aware that the failover command has been run and converts itself into a replication standby location. At that point, replication is again active from the new primary location to the new standby location.

3. The primary location is not currently available and the user does not want to wait until it is back online.

In this scenario, the command verifies the status of the replication primary. If the primary is not accessible and the timeout period has expired (if specified), then the command restores the standby location to its state as of the last successful replication transfer and converts it into a replication primary. Some data loss is possible, for instance if there was a transfer in progress when the primary location became unavailable. After the failover command has been run, the user has two options:

- First, a new standby location can be configured using the `acfsutil repl update` command. Note that it is harmless if the original primary returns (as a standby) after the user has specified the new standby. The original primary will remain idle (as a standby) until the user runs `acfsutil repl terminate standby` for it.
- Alternatively, the user can terminate replication, by running `acfsutil repl terminate primary` on the new primary.

You should bear in mind that until an active standby is in place following failover, your data will not be protected by replication.

When the current primary location is active, you should quiesce application updates to the primary before running `acfsutil repl failover`. Any updates attempted to the current primary location before its conversion to a standby are discarded. Any updates attempted to it after its conversion fail, just as updates would to any other standby location.

When application updates are resumed, they must be directed to the new primary location.

You run the failover command in the standby cluster simply by naming the standby storage location to be used for it:

```
[joe@primary-node1 ~]$ acfsutil repl failover /smntpt
```


Termination of Replication

The “`acfsutil repl terminate`” command is used on both the primary location and the standby location in cases when administrators need to dissolve the instantiated replication. Note that this is performed at the file system level and not at the node level. To perform graceful ACFS Terminate, it is recommended to terminate the primary file system first, followed with by a terminate command at the standby file system.

Note: “`acfsutil repl terminate primary`” does not wait for an in-progress replication operation to complete. Termination is effective immediately.

TERMINATE REPLICATION ON PRIMARY NODE

```
[joe@primary-nodel ~]$ acfsutil repl terminate primary /pmntpnt
```

TERMINATE REPLICATION ON STANDBY NODE

```
[joe@standby-nodel ~]$ acfsutil repl terminate standby /smntpnt
```

Once replication termination has completed for a specific file system on a given Primary and a given Standby, no replication infrastructure exists between that primary and standby file system. Termination of replication is a permanent operation and requires a full re-initialization to instantiate replication anew.

To restart replication, the “`acfsutil repl init`” command is used as previously illustrated.

UPGRADING FROM PRE-12.2 TO SNAPSHOT-BASED REPLICATION

These commands provide support for upgrading an active, pre-existent replication relationship to use snapshot-based replication. The goal of this support is to provide an active snapshot-based replication relationship between the same primary and standby, with no need to terminate or re-initiate replication, and of course no data lost in replicating.

Before beginning the upgrade procedure, the user must ensure that the primary file system is mounted on only one node of his cluster. As noted below, it is also recommended that the user pause any application activity targeting the primary or standby file system, though the upgrade will proceed even if this restriction is not obeyed.

It should also be noted that, once the procedure has been started, it must be completed – there is no provision for aborting the upgrade.

The first step in the procedure is to run the “`acfsutil repl upgrade prepare`” command on the primary cluster. This command specifies the user and host or interface name that will be used for snapshot-based replication, as well as the primary mount point. The user and host names are given with the option `-s`, exactly as they are for the “`acfsutil repl init primary`” command for snapshot-based replication.

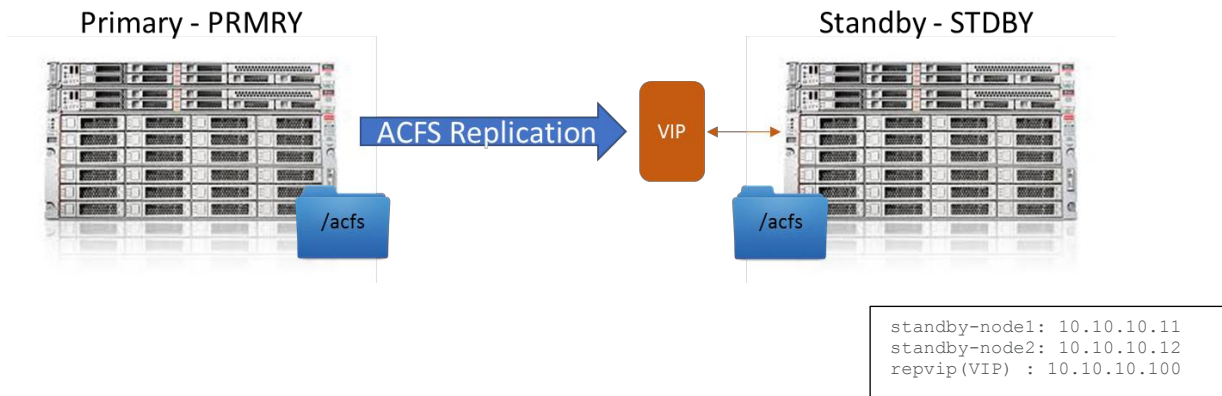
The next step is for the user to upgrade the standby. On the standby, he runs the “`acfsutil repl upgrade standby`” command. This command specifies the user that will be used for snapshot-based replication, as well as the standby mount point. The user name is given with “`-u`”, exactly as it is for the “`repl init standby`” command for snap-based replication. Once this command has been issued, the upgrade must be completed by issuing the “`acfsutil repl upgrade primary`” command.

The final step in the upgrade procedure is to run the “`acfsutil repl upgrade primary`” command on the primary. This is the command that atomically terminates original replication and initiates snap-based replication. This command accepts any of the command-line options accepted by the “`repl init primary`” command for snap-based replication, except for the `-m` option and `-s` option. Instead of using these options, the command obtains the needed information from other sources.

It is recommended that the user quiesce any applications using the primary or standby file systems to be upgraded. This will ensure an optimal upgrade.

APPENDIX A – CASE STUDY

ACFS replication between two Oracle Database Appliance (ODA), through a Virtual IP (VIP) on the standby. Note that the steps described here will set up replication in just one direction – that is, the steps are not sufficient to support failover (as described above).



STEPS

1. Add a VIP configuration to the Oracle Clusterware on Standby ODA (as `root`) using 'appvipcfg':

```
Usage: appvipcfg create -network=<network_number> \
      -ip=<ip_address> \
      -vipname=<vipname> \
      -user=<user_name>[-group=<group_name>] \
      [-failback=0 | 1]

Example:
[root@primary-node1 ~]# /u01/app/12.1.0.2/grid/bin/appvipcfg create \
                        -network=1 \
                        -ip=10.10.10.100 \
                        -vipname=repvip \
                        -user root \
                        -failback=1
```

After you have created the application VIP using this configuration script, you can view the VIP profile using the following command:

```
$GRID_HOME/bin/crsctl status res repvip -p
```

As root, start the VIP resource:

```
[root@primary-node1 ~]# crsctl start resource repvip
```

2. On standby, create the “repluser” on both ODA nodes:

```
[root@standby-node1 ~]# useradd -g asmadmin repluser
[root@standby-node1 ~]# passwd repluser
[root@standby-node1 ~]# mkdir -p /home/repluser/.ssh
[root@standby-node1 ~]# chown repluser:asmadmin /home/repluser/.ssh

[root@standby-node2 ~]# useradd -g asmadmin repluser
[root@standby-node2 ~]# passwd repluser
[root@standby-node2 ~]# mkdir -p /home/repluser/.ssh
[root@standby-node2 ~]# chown repluser:asmadmin /home/repluser/.ssh
```

3. On Primary, on both nodes generate the authorized key (as `repluser`):

```
[repluser@primary-node1 ~]$ ssh-keygen -t rsa
[repluser@primary-node2 ~]$ ssh-keygen -t rsa
```

4. Copy the new keys to the standby:

```
[repluser@primary-node1 ~]$ ssh-copy-id -i ~repluser/.ssh/id_rsa.pub repluser@standby-node1
[repluser@primary-node2 ~]$ ssh-copy-id -i ~repluser/.ssh/id_rsa.pub repluser@standby-node1
[repluser@primary-node1 ~]$ ssh-copy-id -i ~repluser/.ssh/id_rsa.pub repluser@standby-node2
[repluser@primary-node2 ~]$ ssh-copy-id -i ~repluser/.ssh/id_rsa.pub repluser@standby-node2
```

Note that on both STDBY nodes the “authorized_keys” file will be as following:

```
$ cat /home/repluser/.ssh/authorized_keys
```

```
ssh-rsa
AAAAB3NzaC1yc2EAAAABIWAAAQEA4xwODqKntKgq7TWYzWs0G8fsjRdmHUGFCYM5Gw+ibsXwcvXZtfK0ec
7ypmUL3YK4zbUtqKUL4lYPr4iI015RgAPSomZc+vpPPCD9Rbz2dKQPvJH4n6m991fTxxrp96VeUic2yk29xt
MT66V7RmqGrJMfBFuX8w1VaxJ4rLq4QDU/b3+xMAIrx2e6a213H1yqhywgl+CntzCGc8yVWhACLRHgEFaUz
8P5/eg+QM1aLysaTcx7OHhGtHUqADPCJ8mCWuujk1SI2qpcOIZXQr+feIW8X3KnmG4qUzrjPKXMvupur3w
Phbo/z3SvspBDFWN3DiV57len/b21wVpeGjgcQ== repluser@primary-node1

ssh-rsa
AAAAB3NzaC1yc2EAAAABIWAAAQEA2rI13+gdbGIIdmUuqNYoUAnQyvVuKOL4Uj5MR04x/kVOIJIxdjnJhbW5
BM270eln25d4eAbKdMjORnE2AIG3E/av2E2dyLAIJ4+hngPXapgsIvbzG2PBrZ3RaXVv1lq2BBK/t3FH05y
VUj17HQ7gyzDxvRyo652E7994a7BK/AawokWNJ68LDdT8URnCWUnShlwsC2/0ds15nUhgLjTB2A7UOPfT14
F3DkujfxDORPB2blgU7PvOVnIzXcF58oux0W4xcLXyAeOPIK7S/u48PSnFjnKJjdx/pX7OwLFdbECndOKpl
isFmKLMcMI24KUV459YW08mymuPdrjoyA+JF0Q== repluser@primary-node2
```

At this time, you can connect from Primary ODA to Standby ODA as “repluser” with password-less functionality:

```
[repluser@primary-node1 ~]$ ssh repluser@standby-node1 date
Fri Nov 6 11:30:03 SAST 2020
```

```
[repluser@primary-node1 ~]$ ssh repluser@standby-node2 date
Fri Nov 6 11:30:12 SAST 2020
```

```
[repluser@primary-node2 ~]$ ssh repluser@standby-node1 date
Fri Nov 6 11:30:29 SAST 2020
```

```
[repluser@primary-node2 ~]$ ssh repluser@standby-node2 date
Fri Nov 6 11:30:34 SAST 2020
```

The “known_hosts” file on the Primary will be something like:

```
$ cat ~repluser/.ssh/known_hosts
```

```
standby-node1.it.oracle.com,10.10.10.11 ssh-rsa
AAAAB3NzaC1yc2EAAAABIWAAAQEAAsvEtyJXoxUXwL1zIOq9JjbZIBCy0HSw+gvR0Xff+Uk2hxfWWAV82oZ4
qYEykCXU+pJBB0wcTM871tPQuEmaMJ9vJv5GplNVPz3UOZQovxsgjE7yLWolOFT0S4xztm5XhU01x7s_jmGY
I3nAA8vTf17s9Q2AGEXSzfYh0BCWQ9Q7Es5BCEpcLeHjZAmimktjaiXgKp9ABOXOnTYfwJXhRM/5RJ+k4HP
4yRwvEYUKlM3FkZ2uBJNkdaMwV7I+8Lr4MkVwik3V0ukKZ5043YFnji8wyHn1A9kuoqeHKC+eALPA5Lmlfh
r0f2q1/RA1c09j5qWdxBYtWWrP9VnCi195qZfQ==

standby-node2.it.oracle.com,10.10.10.12 ssh-rsa
AAAAB3NzaC1yc2EAAAABIWAAAQEAarkmXmij6Xz57KH0010V7BTT8/rnvHq7PSizqTs/FI9SsePqq3byOKbx
ekik/C185xC3NYqp6577B+c7t36wj0b8vLdlbuMSySOTk6cFiVb87sIJWNW/iHYLdFfjdtkZ1pvR1W6uNRz
R8DryksjiMqgFF8g2k75r08jncdK5yCfSs1RstOfdHjPtrj6c/uabH8Ve8Ehd06IwztwjSqjwuW711U/GT2
680qOFUrZMP+A36GSiYnquanpMwUxvVHswqUkrqEKTJWkd2EaOER1Ftxsecnsdfxw0wG6gjbwxhFGUUCul
OKpURgLdJuAH9Q9RaSBzoYgJ9JYxlimXLQZQ2w==
```

As we need to use the Standby VIP (10.10.10.100) created above, you need to modify “known_hosts” on Primary to read as follows:

```
$ cat ~repluser/.ssh/known_hosts
```

```
repvip.it.oracle.com,standby-node1.it.oracle.com,10.10.10.11,10.10.10.100 ssh-rsa
AAAAB3NzaC1yc2EAAAABIWAAAQEAAsvEtyJXoxUXwL1zIOq9JjbZIBCy0HSw+gvR0Xff+Uk2hxfWWAV82oZ4qYEykCXU+p
JBB0wcTM871tPQuEmaMJ9vJv5GplNVPz3UOZQovxsgjE7yLWolOFT0S4xztm5XhU01x7s_jmGYI3nAA8vTf17s9Q2AGEXS
zFYh0BCWQ9Q7Es5BCEpcLeHjZAmimktjaiXgKp9ABOXOnTYfwJXhRM/5RJ+k4HP4yRwvEYUKlM3FkZ2uBJNkdaMwV7I+8
Lr4MkVwik3V0ukKZ5043YFnji8wyHn1A9kuoqeHKC+eALPA5Lmlfhr0f2q1/RA1c09j5qWdxBYtWWrP9VnCi195qZfQ==
```

```
repvip.it.oracle.com,standby-node2.it.oracle.com,10.10.10.12,10.10.10.100 ssh-rsa
AAAAAB3NzaC1yc2EAAAABIwAAAQEAArkMxmij6Xz57KH0010V7BTT8/rnvHq7PSizqTs/FI9SsePqq3byOKbxekik/C185x
C3NYqp6577B+c7t36wjob8vLdlbuMSySOTk6cFiVb87sIJWNW/iHYLdFfjdtkZ1pvR1W6uNRzR8DryksjiMqgFF8g2k75
r08jnCdK5yCfSslRstOfdHjPtrj6c/uabH8Ve8EhdO6IwztwjSqjwuW711U/GT2680qOFUrZMP+A36GsiYnquanpMwUxv
VWHswqUkrqEKTJWkd2EaOERlFtxsecnsdfxw0wG6gjbwxhFGUUCulOKpURgLdJuAH9Q9RaSBzoYGj9JYxlimXLQZQ2w==
```

5. Validate the ssh setup by issuing the following commands (as *repluser*):

```
[repluser@primary-nodel ~]$ acfsutil repl info -c -u repluser standby-nodel standby-node2 /acfs
A valid 'ssh' connection was detected for standby node standby-nodel as user repluser
A valid 'ssh' connection was detected for standby node standby-node2 as user repluser

[repluser@primary-node2 ~]$ acfsutil repl info -c -u repluser standby-nodel standby-node2 /acfs
A valid 'ssh' connection was detected for standby node standby-nodel as user repluser
A valid 'ssh' connection was detected for standby node standby-node2 as user repluser
```

The commands above should be run on each primary host. Each command line run should name every standby host.

6. Initialize the replication on Standby first:

```
[joe@standby-nodel ~]$ acfsutil repl init standby -u repluser /acfs
```

7. Then initialize the replication on Primary:

```
[joe@primary-nodel ~]$ acfsutil repl init primary -C -s repluser@rep-stdby -m /acfs /acfs
```

8. You can check the replication by issuing:

```
[joe@primary-nodel ~]$ acfsutil repl info /acfs
Time: Fri Nov 06 12:12:09 2020
Event: Replication Initialization Started
Initialization Start Time: Fri Nov 06 12:12:09 2020

Time: Fri Nov 06 12:12:04 2020
Transfer completed at: Fri Nov 06 12:12:04 2020
Transfer status: 0

Time: Fri Nov 06 12:12:04 2020
Event: Clone Created
Session started at: Fri Nov 06 12:12:04 2020
Session ended at: Fri Nov 06 12:12:04 2020
Number of bytes transmitted: 20759

Time: Fri Nov 06 12:12:04 2020
Create completed at: Fri Nov 06 12:12:04 2020
Create Status: 0

Time: Fri Nov 06 12:12:04 2020
Create started at: Fri Nov 06 12:12:04 2020

Time: Fri Nov 06 12:12:04 2020
Transfer started at: Fri Nov 06 12:12:04 2020

Number of Events: 6
```

CONNECT WITH US

Call +1.800.ORACLE1 or visit oracle.com.
Outside North America, find your local office at oracle.com/contact.

 blogs.oracle.com

 facebook.com/oracle

 twitter.com/oracle

Copyright © 2022, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0120

Oracle ACFS – Snapshots-Based Replication - How to Setup Guide
May, 2022

Authors: Ruggero Citton, Don Bolinger

Contributing Authors: Barb Glover, Fred Glover, ACFS Product Development, ACFS Product Management, RAC Pack, Cloud Innovation and Solution Engineering Team

