

TECHNICAL WHITE PAPER

Protect Your Oracle Database in the Cloud

Explore Purity CloudSnap™ Hybrid Recovery for Oracle.

Contents

- Introduction **3**
 - Purpose 3
 - Audience 3
- CloudSnap Overview **3**
 - Key Benefits 4
 - Simplicity 4
 - Efficiency 4
 - Security 4
 - Cost Savings 4
- Protecting Oracle Databases **5**
 - How CloudSnap Changes the Story 5
 - Environment 6
- Setting up Purity CloudSnap **6**
 - Step 1. Install the Offload App 6
 - Step 2. Configure Network Connectivity for the Offload App 7
 - Step 3. Set Up an S3 Bucket 8
 - Step 4. Add the S3 Bucket as New Offload Target 14
- Offloading an Oracle Database to S3 **16**
- Restoring an Oracle Database from S3 **20**
- Conclusion **23**
- Additional Resources **23**
 - Product Information 23

Introduction

One of the leading features that differentiates Pure Storage FlashArray™ from competitors is the Purity Snapshot technology. Irrespective of volume size, you can create Purity Snapshots nearly instantly and they consume very little additional space. Once created, Purity Snapshots are independent entities and do not rely on the existence of any previous snapshots, or even the source volumes from which they were created. And creating numerous snapshots does not result in performance degradation while reading and writing from the source volume.

Before Purity//FA 5.0, you could replicate Purity Snapshots only to another FlashArray. Now, with Pure's portable snapshot technology, you can move Purity Snapshots freely (or "off-load" them) in a space and network bandwidth-efficient manner to FlashBlade™, third-party NFS servers, and the cloud. With Purity CloudSnap™, a FlashArray appliance can now send Purity Snapshots directly to the cloud.

Purpose

The purpose of this technical white paper is to showcase the simplicity and power of Purity CloudSnap and how it can fit in your organization's Oracle database protection and recovery strategy. It will walk through the steps required to set up offloading of Oracle snapshot backups from a FlashArray to AWS S3 storage using Purity CloudSnap, as well as restoring it back on another FlashArray.

Audience

The target audience for this document includes, but is not limited to, Oracle database administrators, storage administrators, IT managers, and others who are interested in learning more about [Purity CloudSnap](#) and how it can complement their existing on-premises Oracle data protection and recovery strategy. A working knowledge of Oracle, Linux, AWS, server, storage, and networks is assumed but is not a prerequisite to read this document.

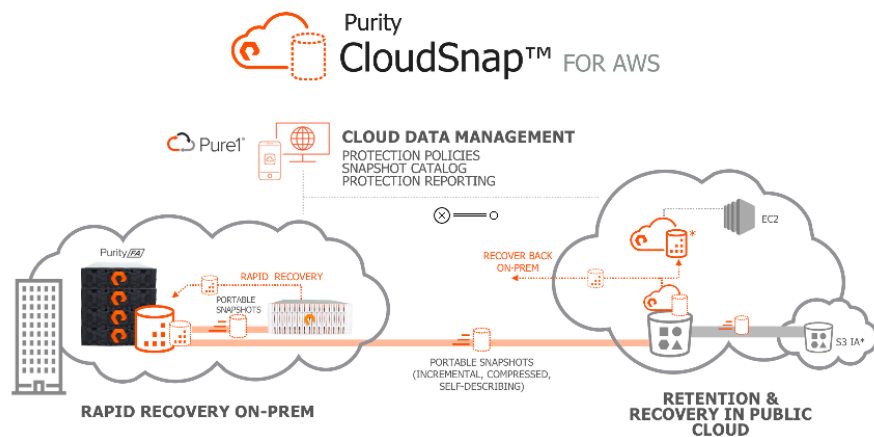
CloudSnap Overview

Purity CloudSnap provides simple, built-in ["self-backup to cloud" protection](#) for FlashArray appliances using Pure's portable snapshot technology. The portable snapshot technology encapsulates both metadata and data into the snapshot, making it portable so it can be offloaded from a Pure FlashArray to the cloud in a format that is recoverable to any FlashArray or to a Pure Cloud Block Store virtual FlashArray appliance.

Pure first introduced its portable snapshot technology via the Snap-to-NFS feature in Purity//5.1, which enabled a FlashArray to offload snapshots to a Pure FlashBlade and to any other NFS server. Pure has extended this technology to enable data protection in the cloud.



- In Purity//FA 5.2, CloudSnap was introduced to offload snapshots directly to an AWS S3 bucket.
- In Purity//FA 5.3, Pure added support for AWS S3 IA (infrequent access) as well as offloading snapshots to Microsoft Azure Blob. (This paper primarily focuses on AWS S3, but the concepts are the same and would apply to Azure Blob.)



Pure portable snapshot technology is deeply integrated with the [Purity Operating Environment](#). It preserves data compression on the wire and in the cloud, thereby saving network bandwidth as well as storage space.

Pure portable snapshots preserve data reduction across snapshots of a volume. After the first snapshot of a volume is offloaded, it becomes the baseline and subsequent snapshots only send delta changes. The snapshot-differencing engine that runs within the Purity Operating Environment uses the local copy of the snapshot to compute the delta changes. Because this drastically reduces the network round trips that would otherwise be required, it reduces network congestion and provides data-access cost savings.

When it comes to restore, Purity knows which blocks already exist on the [FlashArray](#), so it only needs to pull back missing blocks to rebuild the complete snapshot on the FlashArray.

Key Benefits

Simplicity

- Once configured as an offload target in FlashArray, the S3 bucket appears like any other replication target.
- CloudSnap is natively managed via Pure FlashArray's GUI, CLI, and REST interfaces.
- CloudSnap is integrated with the Pure1® Snapshot Catalog and provides users with a global view of all their snapshots.

Efficiency

- CloudSnap requires less space consumption in the S3 bucket, reduces network utilization, and requires smaller backup windows.

Security

- Data is sent to the cloud over https: and stored in an encrypted format in the S3 bucket.

Cost Savings

- Because CloudSnap is built into Purity, you don't need to buy additional backup software or hardware, nor use an additional management interface.



- CloudSnap is aware of the cost-model of each public-cloud platform and makes cost-model-based decisions to reduce infrastructure costs. For example, in the case of AWS, if a snapshot needs to be stored for more than 30 days, CloudSnap would automatically store it in S3 IA, which is considerably cheaper.

Protecting Oracle Databases

The Oracle Recovery Manager (RMAN) utility is built into the Oracle database software to automate the backup, restore, and recovery of an Oracle database. It was introduced in Oracle 8 and Oracle adds new features with every new release. Unique features include block-level corruption detection and repair, plus tight integration with Oracle Enterprise Manager.

RMAN has typically played a central role in an organization's Oracle database protection strategy because it offers a rich set of capabilities, it's platform agnostic, and it is designed, recommended, and supported by Oracle.

However, a gradual shift is happening. Because of their exceptional capabilities, FlashArray snapshots are playing an increasingly important role in the Oracle database protection and recovery strategies of Pure Storage customers. Some organizations base their entire Oracle database protection strategy—including mission-critical databases—on FlashArray snapshots. The primary reasons they cite for using Purity Snapshots as the backbone of their [data-protection strategy](#) include the following:

- Easier and faster to take a snapshot-based backup
- Quicker point-in-time recovery using FlashArray Snapshots and archive log files
- Savings in storage due to space efficiency of FlashArray snapshots
- Lower CPU utilization on the database host due to not having to run RMAN
- Possible reduction in Oracle license cost due to lower CPU utilization
- Purity ActiveCluster™ removes the FlashArray as the single point of failure.

In addition, the stability of Oracle code is a factor. Because database corruption events are very rare, you can do periodic snapshot validation and database corruption checks by hydrating the database backup on a different server and running database-validation commands.

This is not by any means a suggestion against using Oracle RMAN. However, not all snapshots are the same and Purity Snapshot technology is enabling solutions and strategies that are “outside the box.” You should do your own cost and risk assessment. Decide on a data-protection strategy that takes into account your business requirements and the pros and cons of each solution.

How CloudSnap Changes the Story

It is well known that snapshots are not the same as backups. That's true whether they are stored on the same FlashArray or another FlashArray in the same failure domain. This white paper shows how CloudSnap changes that.

FlashArray has had the capability to asynchronously replicate snapshots to a remote FlashArray for many years. This enabled you to store snapshots externally from the source array and provides a simple, flexible, and efficient low-RPO solution for data protection and disaster recovery (DR). However, this approach required you to have a second FlashArray at the remote site. Moreover, it didn't have the agility and economics of the cloud. That has changed with CloudSnap.



Environment

Component	Type
Database	Oracle Enterprise Edition 18c
Host OS	Linux
Cloud Target	AWS S3
On-premises storage	FlashArray//X 90R2

Setting up Purity CloudSnap

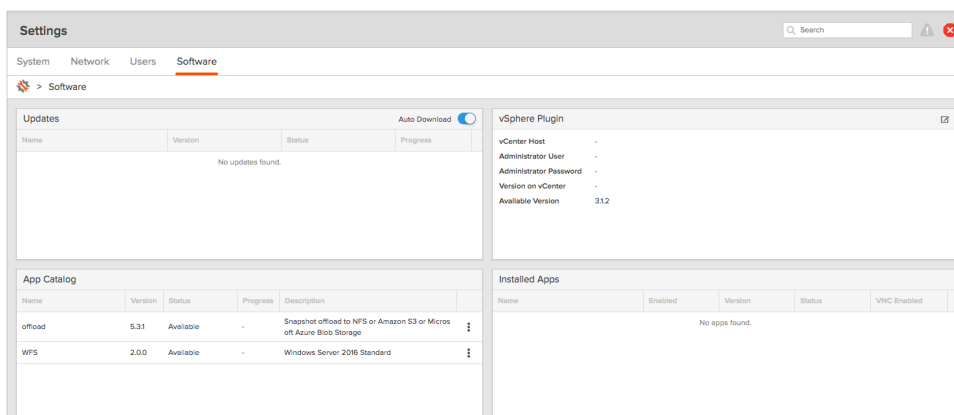
First, verify the following pre-requisites:

- Make sure Purity version is at least 5.2.
- Make sure you have credentials for an AWS account that can create an S3 bucket

Here are the high-level steps that you need to perform:

1. Install the offload app
2. Configure network connectivity for the offload app
3. Set up a S3 bucket
4. Add the S3 bucket as new Offload Target

Step 1. Install the Offload App



Once install is complete, it will show up under **Installed Apps** as shown below.

Installed Apps					
Name	Enabled	Version	Status	Description	
offload	true	5.2.3	healthy	Snapshot offload to NFS or Amazon S3	⋮

Step 2. Configure Network Connectivity for the Offload App

Create a virtual interface (vif) for the Offload app and select a physical interface on the FlashArray to connect to the vif. (Make sure that the physical interface has network connectivity to the offload target.) If a "replbond" interface exists on the FlashArray, connect the vif for the Offload app to the replbond interface.

```
pureuser@sn1-x70-f05-27> purenetwork create vif @offload.data0 --subinterfacelist replbond
```

Otherwise, if replbond has not been created, connect the vif for the Offload app directly to a replication port on the FlashArray.

```
pureuser@sn1-x70-f05-27> purenetwork create vif @offload.data0 --subinterfacelist
ct0.eth3,ctl1.eth3
```

Next, assign an IP to the virtual interface:

```
pureuser@sn1-x70-f05-27> purenetwork setattr --address 10.21.228.16 --netmask 255.255.255.0 --
gateway 10.21.228.1 @offload.data0
```

Now, enable the interface:

```
pureuser@sn1-x70-f05-27> purenetwork enable @offload.data0
```

Finally, restart the Offload App:

```
pureuser@sn1-x70-f05-27> pureapp disable offload
pureuser@sn1-x70-f05-27> pureapp enable offload
```

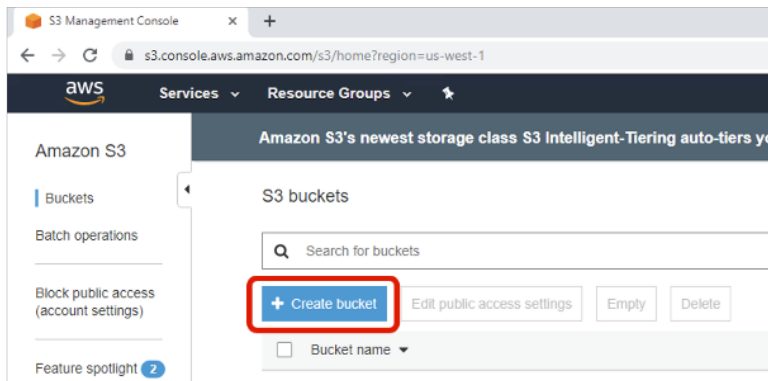
Use the list command to make sure that the status of the Offload app is healthy.

```
pureuser@sn1-x70-f05-27> pureapp list
Name      Enabled  Version  Status  Description                                Details
offload   True     5.3.0    healthy Snapshot offload to NFS or Amazon S3    -
```

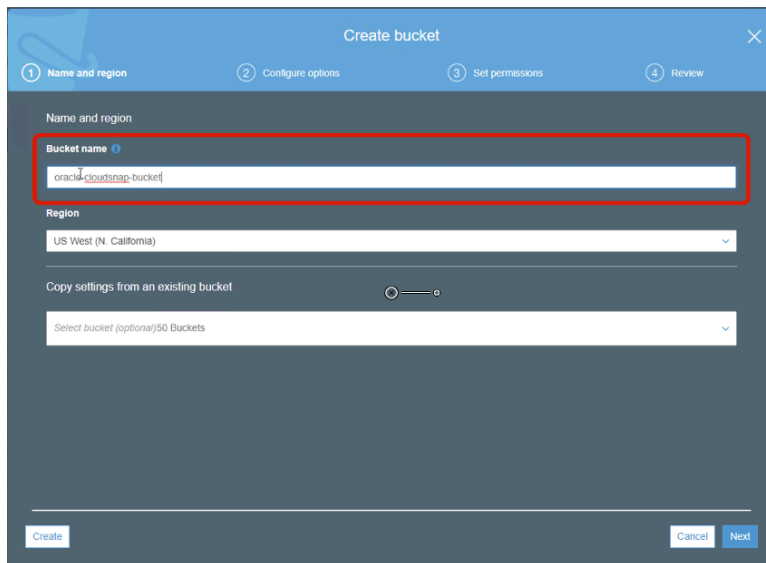


Step 3. Set Up an S3 Bucket

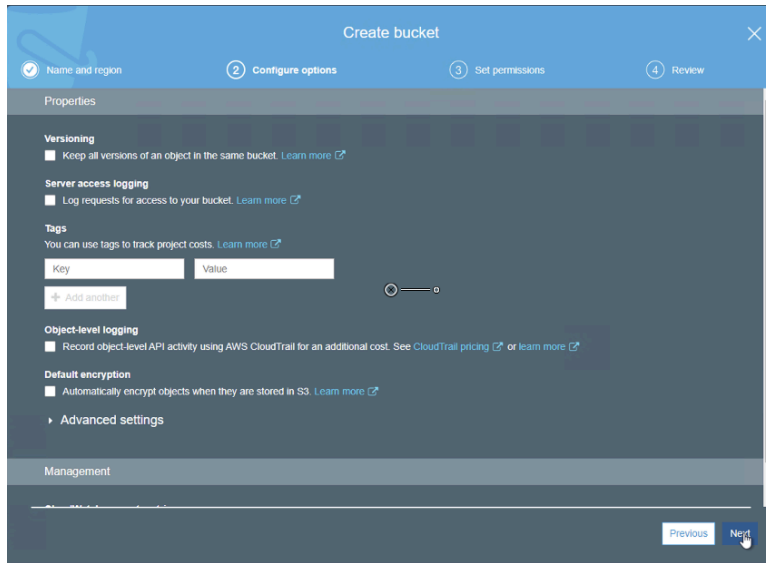
Login to AWS and navigate to the S3 service page. Click on the Create Bucket button.



The **Create Bucket** dialog will pop up. Provide a name for the bucket.

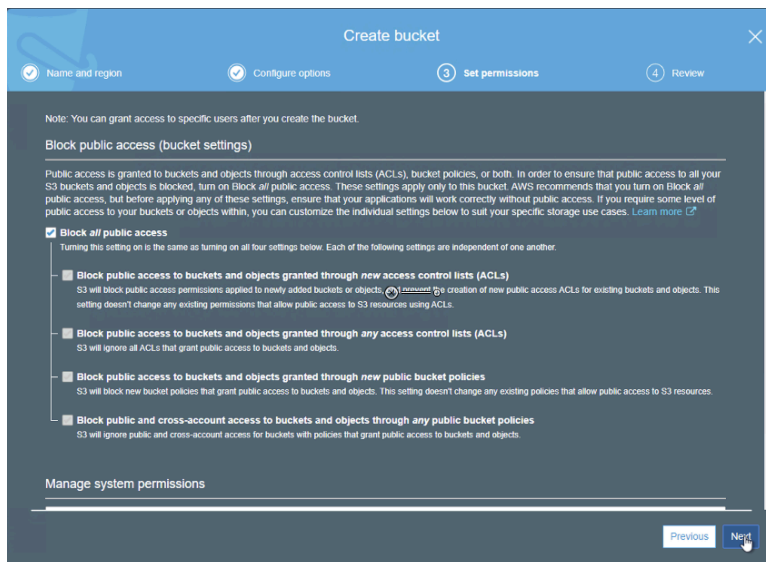


On the **Configure Options** screen, the defaults are good, so move to the next screen.



The screenshot shows the 'Create bucket' wizard in the AWS Management Console, specifically the 'Configure options' step. The progress bar at the top indicates four steps: 1. Name and region (completed), 2. Configure options (current), 3. Set permissions, and 4. Review. The 'Properties' section is expanded, showing various settings. The 'Block all public access' checkbox is checked. Below it, there are four sub-settings, all of which are also checked: 'Block public access to buckets and objects granted through new access control lists (ACLs)', 'Block public access to buckets and objects granted through any access control lists (ACLs)', 'Block public access to buckets and objects granted through new public bucket policies', and 'Block public and cross-account access to buckets and objects through any public bucket policies'. The 'Management' section is partially visible at the bottom. At the bottom right, there are 'Previous' and 'Next' buttons.

Make sure the **Block all public access** checkbox is checked.

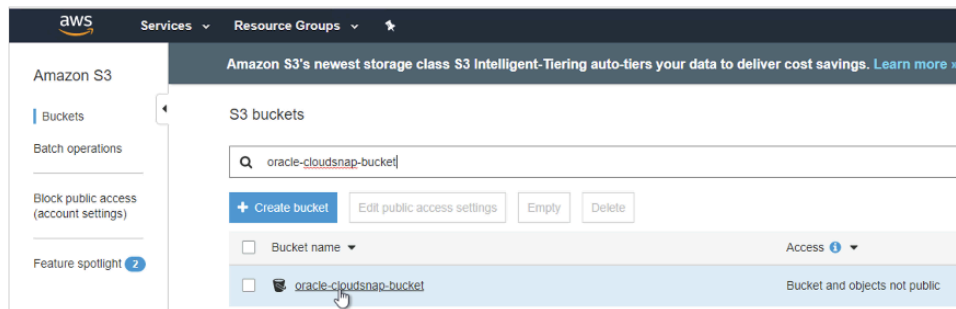


The screenshot shows the 'Create bucket' wizard in the AWS Management Console, specifically the 'Set permissions' step. The progress bar at the top indicates four steps: 1. Name and region (completed), 2. Configure options (completed), 3. Set permissions (current), and 4. Review. The 'Block public access (bucket settings)' section is expanded, showing a note and four sub-settings, all of which are checked. The 'Manage system permissions' section is partially visible at the bottom. At the bottom right, there are 'Previous' and 'Next' buttons.

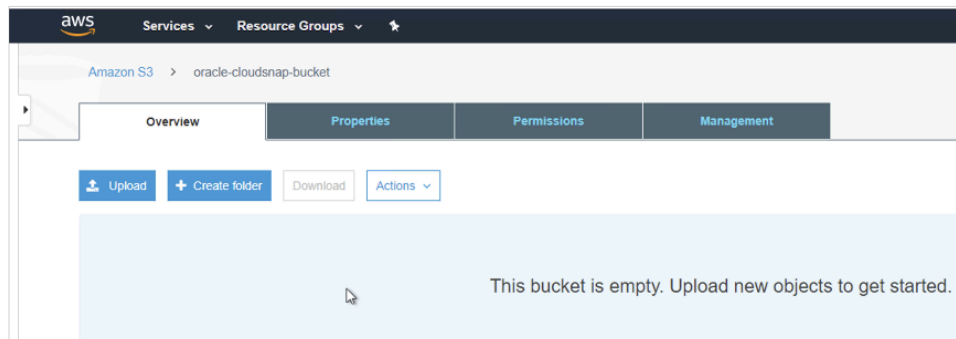
Review the selected options and click on **Create Bucket**.

The screenshot shows the 'Create bucket' wizard in the AWS Management Console, specifically the 'Review' step. The wizard has four steps: 'Name and region', 'Configure options', 'Set permissions', and 'Review' (the current step). The 'Name and region' section shows the bucket name 'oracle-cloudsnap-bucket' and the region 'US West (N. California)'. The 'Options' section lists various settings: Versioning (Disabled), Server access logging (Disabled), Tagging (0 Tags), Object-level logging (Disabled), Default encryption (None), CloudWatch request metrics (Disabled), and Object lock (Disabled). The 'Permissions' section shows 'Block all public access' set to 'On' for both new and existing ACLs. At the bottom right, there are 'Previous' and 'Create bucket' buttons.

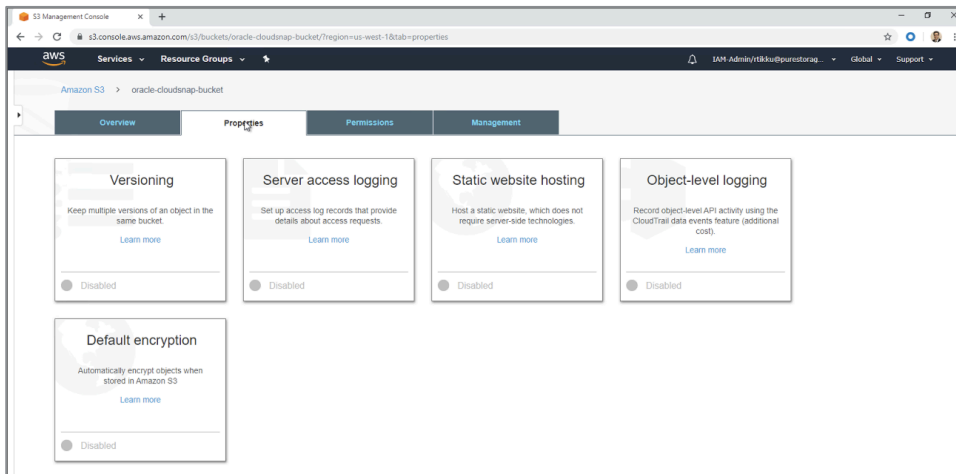
The bucket gets created. Click on it to go to the Bucket details page.



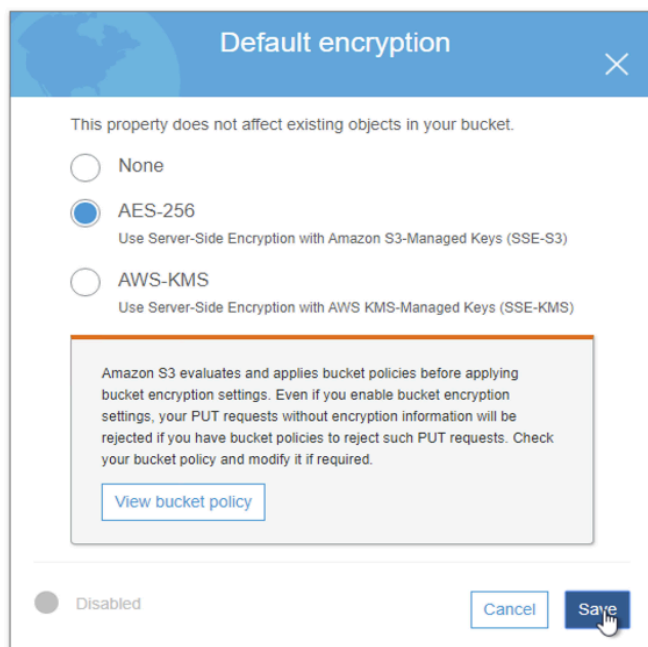
Click on the **Properties** tab.



Click on the Default Encryption tile.



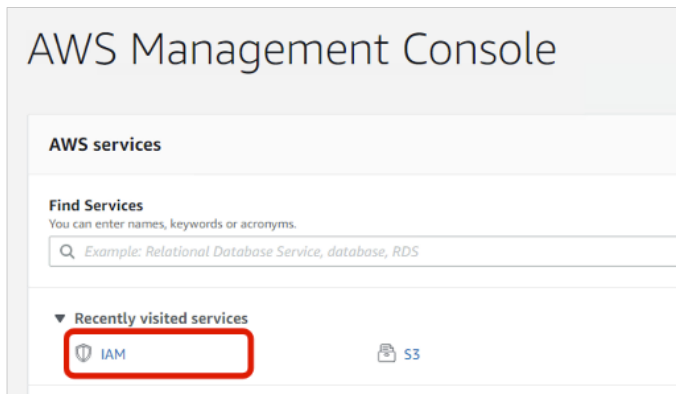
Once expanded, select the AES-256 radio button and hit Save.



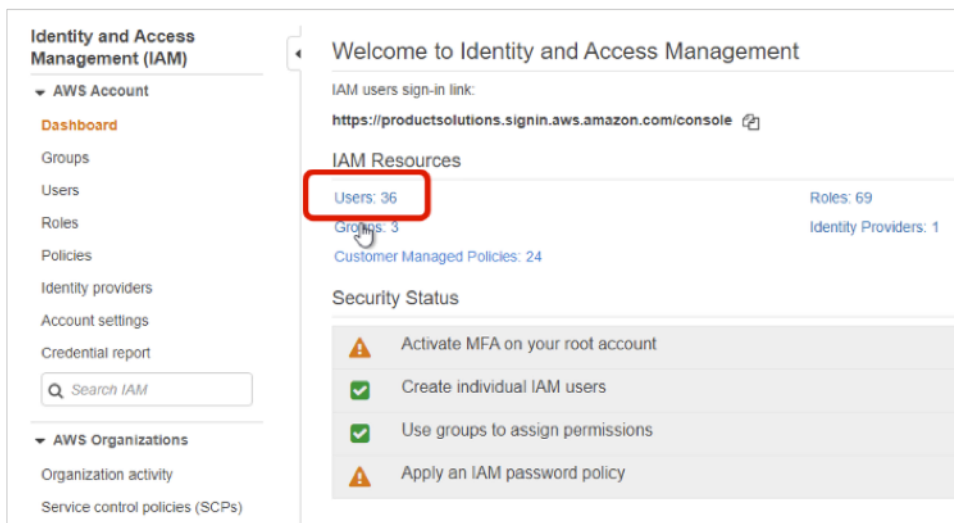
No changes are needed on the remaining tabs. The bucket is ready to use. Now we need to create a User.



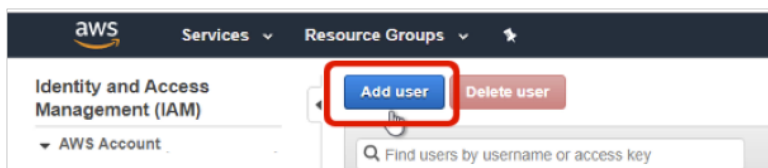
Go to the AWS Services home page and click on the **IAM** service.



Then click on **Users**.



Click on the **Add User** button.



Provide a user name and select the **Programmatic access** checkbox.

Add user

12345

Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name*

[+ Add another user](#)

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Access type* ☒ **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

☐ **AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

Select Attach AmazonS3FullAccess policy to the user.

Add user

12345

Set permissions

Add user to group
Copy permissions from existing user
Attach existing policies directly

Create policy

Filter policies Showing 1 result

	Policy name	Type	Used as	Description
☒	AmazonS3FullAccess	AWS managed	Permissions policy (15)	Provides full access to all buckets via the A...

Add Tags if desired.

Add user

12345

Add tags (optional)

IAM tags are key-value pairs you can add to your user. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this user. [Learn more](#)

Key	Value (optional)	Remove
Add new key		

You can add 50 more tags.



Review inputs provided before selecting **Create User**.

Add user

1 2 3 **4** 5

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name oracleCloudSnap

AWS access type Programmatic access - with an access key

Permissions boundary Permissions boundary is not set

Permissions summary

The following policies will be attached to the user shown above.

Type	Name
Managed policy	AmazonS3FullAccess

Tags

No tags were added.

Finally, the user is created. The confirmation screen provides the Access Key ID and Secret access key. There is also a link to download this information as a csv file.

Add user

1 2 3 4 **5**

Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: <https://productsolutions.signin.aws.amazon.com/console>

[Download .csv](#)

User	Access key ID	Secret access key
oracleCloudSnap	AKIAQX5I	***** Show

Be sure to either click on the Download .csv file or copy the Access key ID and the Secret access key at this point. Once this screen is closed, there is no way to get the security credentials for the IAM user. Save the security credentials and use them when connecting FlashArray to the S3 bucket.

Step 4. Add the S3 Bucket as New Offload Target

Offload Targets

Name	Status	Protocol	Details
No offload targets have been connected.			



Connect Offload Target

Protocol

s3

Web Service

☒ AWS

☐ Other

Name

rtikku-oracle-s3

Access Key ID

AKIAQX5UTL*****

Secret Access Key

Bucket

oracle-cloudsnap-bucket

☒ Initialize bucket as offload target

Placement Strategy

retention-based

Cancel

Connect

Offload Targets				
Name	Status	Protocol	Details	
rtikku-oracle-s3	connected	s3	Bucket: rtikku-oracle-bucket	Access Key ID: AKIAQX5***** Secret Access Key: ****

With this, setting up of CloudSnap is complete on FlashArray sn1-x70-f05-27.



Storage Search Alerts Close

Array Hosts Volumes Protection Groups Pods

> Array

Size: 4348703 G | Data Reduction: 14.0 to 1 | Volumes: 128 T | Snapshots: 155.28 M | Shared: 2.89 T | System: 0.00 | Total: 4.17 T

sn1-x70-f05-27 ID: 6c1d7213-605f-4920-a7b2-b84f7c5b1f91

Hosts	Host Groups	Volumes	Volume Snapshots	Volume Groups	Protection Groups	Protection Group Snapshots	Pods
14	2	135	25	1	15	3	4

Connected Arrays + :

Name	Connected	Type	Version	Management Address	Replication Address	Throttled	
sn1-x70-f05-33	True	sync-replication	5.2.3			False	✓ x
CBS-Oracle	True	async-replication	5.2.0.beta5	10.0.3.136	10.0.3.50	False	✓ x

Offload Targets :

Name	Status	Protocol	Details
rtikku-oracle-s3	connected	s3	Bucket: rtikku-oracle-bucket Access Key ID: AKIAQX5UTL ***** Secret Access Key: **** x

Offloading an Oracle Database to S3

Identify which volumes the Oracle database is sitting on. If it hasn't already been done, create a Protection Group and that contains these volumes.

To create a Protection Group, click on the Protection Groups tab on the Storage page and click on the **plus sign** on the table titled Protection Groups. In the popup dialog, enter the name of the Protection Group and click on **Create**.

Create Protection Group ✕

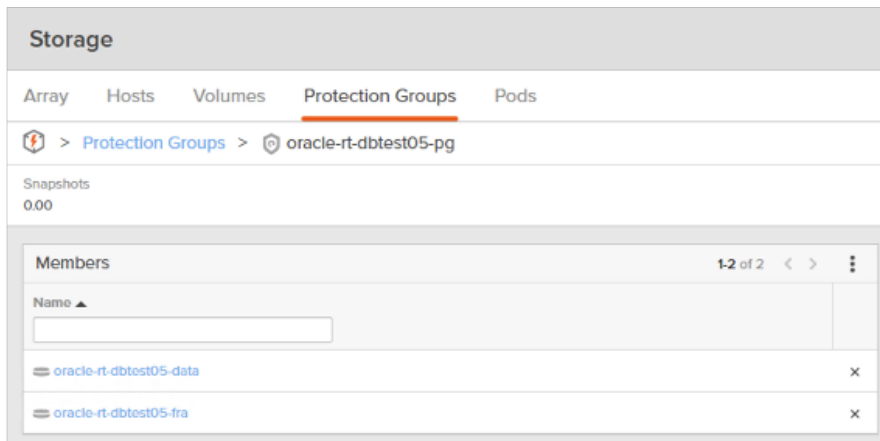
Container:

Name:

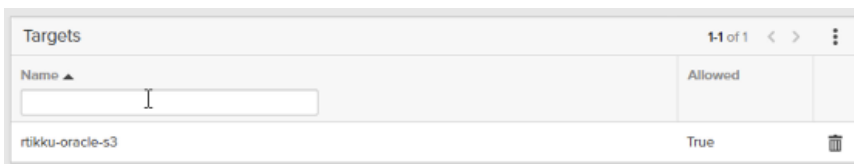
Cancel Create



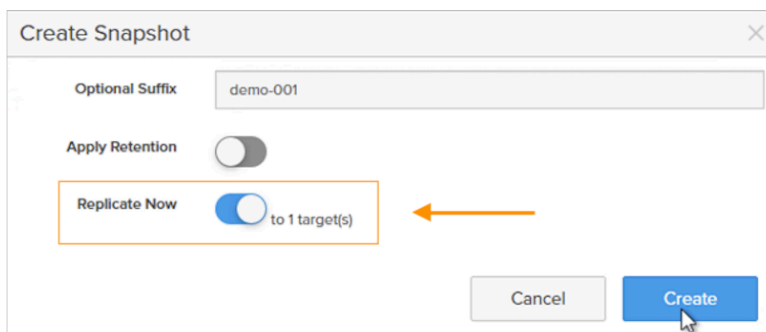
Once created, drill down on this Protection Group to see the Members. Make sure that the Members table lists all the volumes of the database being protected.



Add the S3 offload target created in the previous step to this Protection Group.



Next, create a snapshot of this Protection Group. You can provide an optional suffix to make it easier to identify this Protection Group Snapshot. It is very important to select the **Replicate Now** option, otherwise the snapshot will not be replicated to the target.



Important note: The snapshot taken above is a crash-consistent snapshot. You can use it to restore or clone the Oracle database to the point in time when the snapshot was taken. If the database is in Archive Log mode, and has the ability to do a point-in-time recovery, then you should put the source database in Hot Backup mode **before** creating the snapshot. Do this by executing the following command on the source database: "ALTER DATABASE BEGIN BACKUP"

After the snapshot is created (typically in a matter of seconds), you need to take the database needs out of Hot Backup mode. Do this by executing the following command on the source database: "ALTER DATABASE END BACKUP"



The snapshot taken above was an ad-hoc or on-demand snapshot. When using CloudSnap as part of a larger data-protection strategy, you would schedule snapshot offloading to the cloud.

If a crash-consistent snapshot is all you need, you can schedule snapshot replication to offload target using the GUI itself using the steps given below. However, if you want the ability to use the snapshots for point-in-time recovery, then you need to put the database in Hot Backup mode before taking the snapshot, and take it off Hot backup mode after the snapshot is taken. This is not possible using the GUI method and has to be orchestrated via scripts using the command line or the REST interface.

Go to Storage > Protection Group and select the protection group to bring up the following screen. Click on the edit icon on the right of Replication Schedule.

Replication Schedule

Enabled: **False**

Replicate a snapshot to targets every **4** hours

Retain all snapshots on targets for **1** days

then retain **4** snapshots per day for **7** more days

The following dialog will appear.

Edit Replication Schedule

☐ Disabled

Replicate a snapshot to targets every hours at

except between and

Retain all snapshots on targets for days

then retain snapshots per day for more days

Cancel

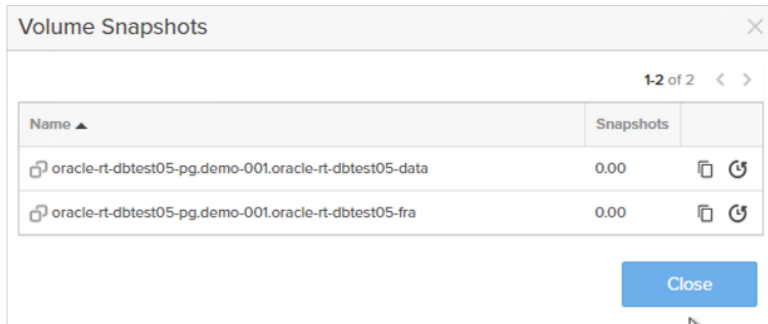
Save

Select the desired replication schedule as well as how long would you like to retain the snapshots on the targets. The Protection Group Snapshots table will now show an entry for the snapshot taken.

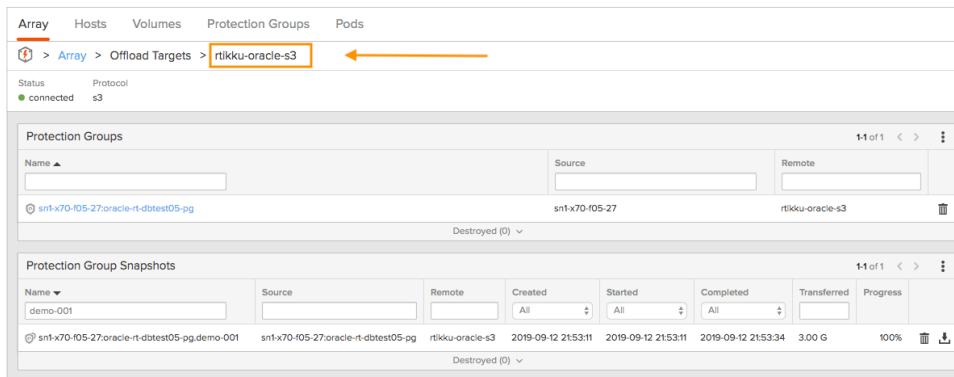
Protection Group Snapshots			1:1 of 1	<	>	+	:
Name	Created	Snapshots					
oracle-rt-dbttest05-pg.demo-001	2019-11-07 19:37:13	0.00					
Destroyed (0)							



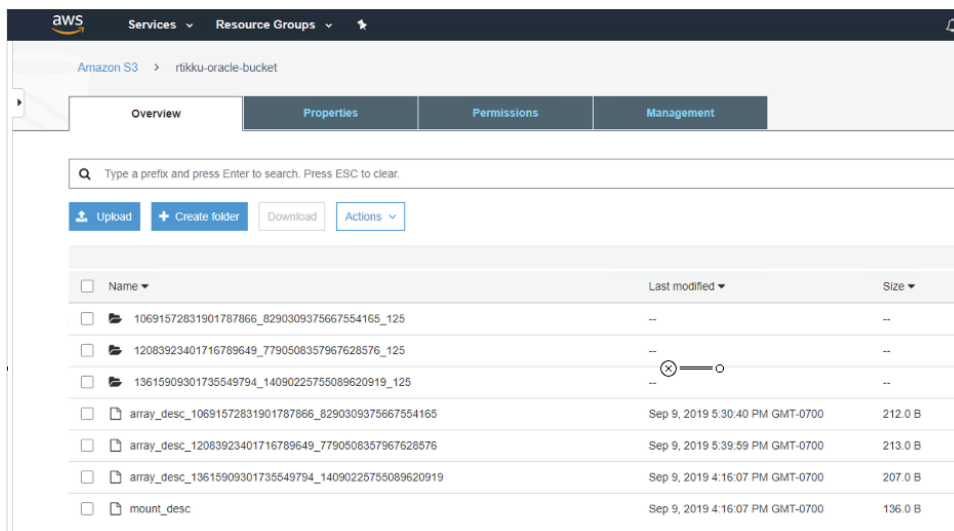
Click on the Protection Group Snapshot to verify that it contains all the volumes for the database.



In the Array tab of the Storage page, click on the S3 bucket hyperlink to verify that the Protection Group has been offloaded to the target. Pay attention to the Progress column: It should show 100%. Also, the **Transferred** column shows the amount of data that was actually transferred, and you can use the **Started** and **Completed** columns to find out the time taken for the transfer.



The AWS S3 details page shows objects that were created in the S3 bucket. However, the data is in an encrypted and proprietary format and can only be restored using the Purity offload interface.



Restoring an Oracle Database from S3

In the previous section, we offloaded protection group **oracle-rt-dbtest05-pg** for an Oracle database **dbtest05** using FlashArray **sn1-x70-f05-27** for storage to an AWS S3 bucket **rtikku-oracle-s3**. Let's now look at the steps to restore the database. You can restore the database back to the same array, to another FlashArray, or to Cloud Block Store.

The following scenario shows how to restore a snapshot from the S3 target to a different FlashArray **sn1-x70-f05-33**. The first step is to install and configure the Offload app as detailed earlier. After the Offload app is installed, you need to add the previously created S3 bucket (that contains the CloudSnap snapshot) as an **Offload Target**.

The screenshot displays the 'Storage' management console. At the top, there's a search bar and a notification icon. Below the header, the 'Array' section shows details for 'sn1-x70-f05-33' with ID '730d1874-06c1-4775-9460-18d81fb926da'. A dashboard of icons shows counts for various resources: Hosts (14), Host Groups (3), Volumes (98), Volume Snapshots (18), Volume Groups (6), Protection Groups (11), Protection Group Snapshots (3), and Pods (5).

The 'Connected Arrays' section contains a table with the following data:

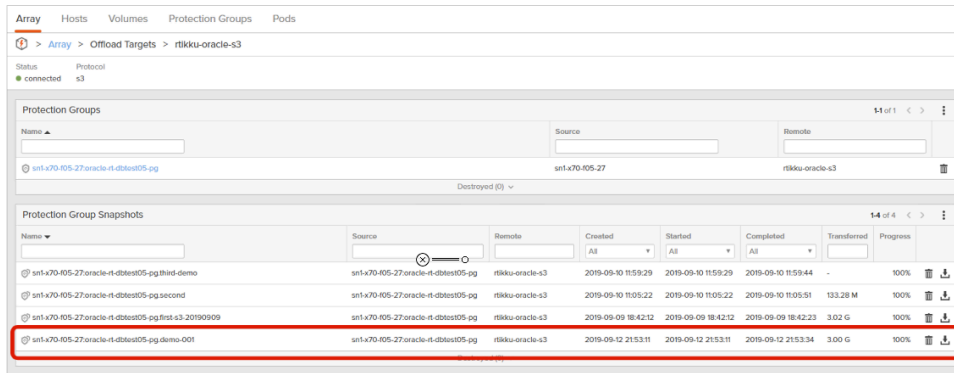
Name	Connected	Type	Version	Management Address	Replication Address	Throttled	
sn1-x70-f05-27	True	sync-replication	5.2.3	10.21.228.24	10.21.228.69 10.21.228.70 10.21.228.71 10.21.228.72	False	☒ x

The 'Offload Targets' section contains a table with the following data:

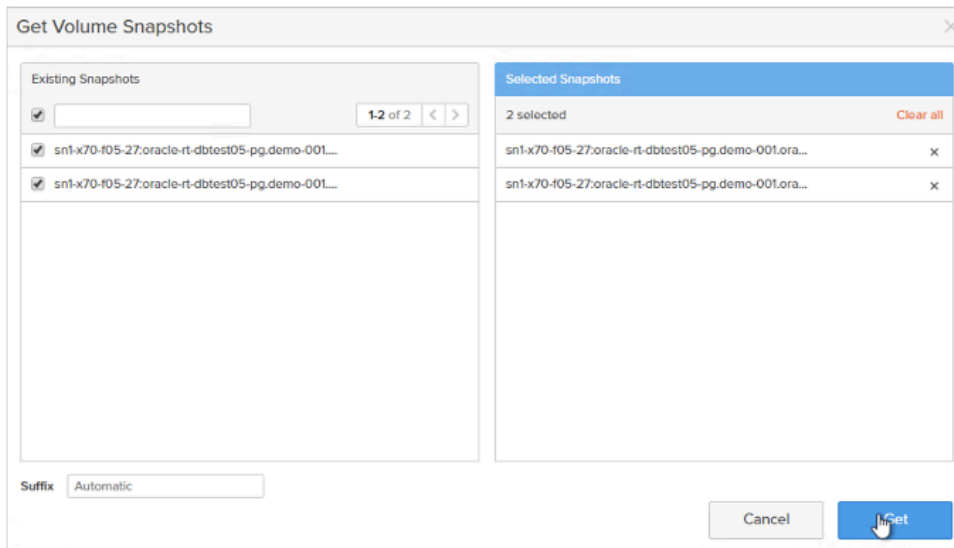
Name	Status	Protocol	Details	
rtikku-oracle-s3	connected	s3	Bucket: rtikku-oracle-bucket Access Key ID: AKIAQX5UTL ***** Secret Access Key: ****	x



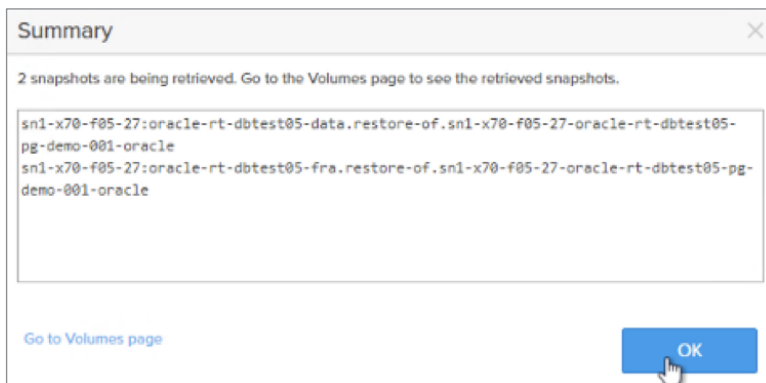
After you've added the S3 Offload Target, click on it to see the Protection Group snapshot backups it contains.



Find the Protection Group snapshot that you need to restore. Click on the **Download** icon in the last column on the right. The following dialog will appear and lists the volumes available in the Protection Group snapshot. Select the database volumes and click on the **Get** button.



Another popup window will appear with confirmation that the volume snapshots have been restored.



Please note that these are *volume snapshots*, not *volumes*. You need to go to the Volumes tab, and create volumes from the volume snapshots.

Click on the **Go to Volumes page** link on the lower left corner of the above popup or go to **Storage > Volumes**. Locate the volume snapshots that were restored from the S3 CloudSnap backup. Click on the three vertical dots icon on the right for each volume that you need to restore.

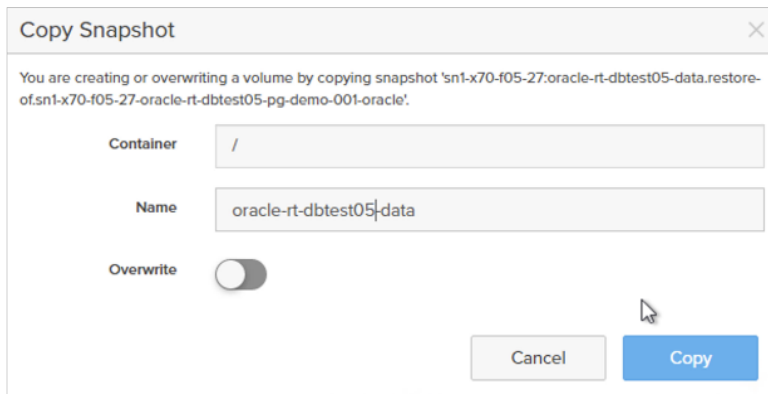
Volume Snapshots			
General Transfer		1-10 of 27	
Name	Created	Snapshots	
sn1-x70-f05-27:pg-demo-asm.3790.dg_oradata01_prod	2019-09-12 21:57:00	0.00	⋮
sn1-x70-f05-27:pg-demo-asm.3790.dg_oradata02_prod	2019-09-12 21:57:00	0.00	⋮
sn1-x70-f05-27:pg-demo-asm.3790.dg_oradata03_prod	2019-09-12 21:57:00	0.00	⋮
sn1-x70-f05-27:pg-demo-asm.3790.dg_oradata04_prod	2019-09-12 21:57:00	0.00	⋮
sn1-x70-f05-27:pg-demo-asm.3790.dg_oradata05_prod	2019-09-12 21:57:00	0.00	⋮
sn1-x70-f05-27:pg-demo-asm.3790.dg_oraredo01_prod	2019-09-12 21:57:00	0.00	⋮
sn1-x70-f05-27:pg-demo-asm.3790.dg_oraredo02_prod	2019-09-12 21:57:00	0.00	⋮
sn1-x70-f05-27:oracle-rt-dbttest05-data.restore-of.sn1-x70-f05-27-oracle-rt-dbttest05-pg-demo-001-oracle	2019-09-12 21:53:11	0.00	⋮
sn1-x70-f05-27:oracle-rt-dbttest05-fra.restore-of.sn1-x70-f05-27-oracle-rt-dbttest05-pg-demo-001-oracle	2019-09-12 21:53:11	0.00	⋮
sn1-x70-f05-27:pg-demo-asm.3789.dg_oradata01_prod	2019-09-12 20:57:00	839.01 K	⋮
Destroyed (0)			

Select **Restore** from the popup menu.

sn1-x70-f05-27:pg-demo-asm.3790.dg_oradata05_prod	2019-09-12	Copy... Restore Rename... Destroy...
sn1-x70-f05-27:pg-demo-asm.3790.dg_oraredo01_prod	2019-09-12	
sn1-x70-f05-27:pg-demo-asm.3790.dg_oraredo02_prod	2019-09-12	
sn1-x70-f05-27:oracle-rt-dbttest05-data.restore-of.sn1-x70-f05-27-oracle-rt-dbttest05-pg-demo-001-oracle	2019-09-12 21:53:11	
sn1-x70-f05-27:oracle-rt-dbttest05-fra.restore-of.sn1-x70-f05-27-oracle-rt-dbttest05-pg-demo-001-oracle	2019-09-12 21:53:11	



This brings up the familiar Copy Snapshot dialog. Provide a new name for the restored volume or use the original name. If the volume already exists, you must select the overwrite option.

A screenshot of a 'Copy Snapshot' dialog box. The title bar says 'Copy Snapshot' with a close button. Below the title bar, there is a message: 'You are creating or overwriting a volume by copying snapshot 'sn1-x70-f05-27:oracle-rt-dbtest05-data.restore-of.sn1-x70-f05-27-oracle-rt-dbtest05-pg-demo-001-oracle''. Below this message, there are three fields: 'Container' with a text input containing '/', 'Name' with a text input containing 'oracle-rt-dbtest05-data', and 'Overwrite' with a toggle switch that is currently turned off. At the bottom right, there are two buttons: 'Cancel' and 'Copy'.

Once you've copied all the volumes belonging to the Oracle Database from the snapshot, the database is ready to be opened after you've performed instance or media recovery.

Conclusion

CloudSnap enables flexible and efficient off-premises [data protection/disaster recovery](#) use cases. Purity Snapshots are playing an increasingly larger role in our customer's Oracle database protection and recovery strategy, and with CloudSnap, these snapshots can be stored on low cost cloud storage for longer term retention.

The Oracle database snapshot backup can not only be restored on any on-premises FlashArray, they can also be restored to a virtual FlashArray in the cloud, known officially as the Cloud Block Store.

Additional Resources

- Learn about [Effortless Cloud Data Protection with CloudSnap](#).
- Download the [CloudSnap / Snap to NFS Installation and Configuration Guide](#).
- Explore [Purity CloudSnap Setup and Best Practices](#).

Product Information

- [Purity CloudSnap](#)
- [Pure Cloud Block Store](#)



© 2019 Pure Storage, Inc. All rights reserved. Pure Storage, Pure1, Pure1 Meta, Pure-On-The-Go, the P Logo, AIRI, the AIRI logo, CloudSnap, DirectFlash, Evergreen, FlashBlade and FlashStack, and ObjectEngine are trademarks or registered trademarks of Pure Storage, Inc. in the U.S. and other countries. All other trademarks are registered marks of their respective owners.

The Pure Storage products and programs described in this documentation are distributed under a license agreement restricting the use, copying, distribution, and decompilation/reverse engineering of the products. No part of this documentation may be reproduced in any form by any means without prior written authorization from Pure Storage, Inc. and its licensors, if any. Pure Storage may make improvements and/or changes in the Pure Storage products and/or the programs described in this documentation at any time without notice.

THIS DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. PURE STORAGE SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

Pure Storage, Inc.
650 Castro Street, #400
Mountain View, CA 94041

PS1040-01 12/2019

purestorage.com

800.379.PURE

