PURESTORAGE®

TECHNICAL WHITE PAPER

# Protect Application Data with ActiveDR in Purity FA//6

Minimize data loss and accelerate response to outages.

# Contents

## Introduction

ActiveDR delivers continuously active replication, near-zero recovery point objective (RPO), and simple disaster recovery (DR) in the Pure Storage® Purity//FA 6 operating environment. ActiveDR seamlessly protects your application data with geo-distance resiliency, minimizing data loss and accelerating response to business outages.

Pure Storage built ActiveDR with a focus on simplifying DR workflows and the need to continuously protect application data. Workflows such as test failover, real failover, resync and failback can all be performed easily and testing failover does not require stopping replication. With ActiveDR, you can protect multiple applications with a single integrated protection strategy.

Continuous replication with ActiveDR provides support for much lower RPOs than traditional array-based replication that periodically performs snapshot-differencing to drive replication. ActiveDR forks the incoming write stream, applies data reduction, and continuously replicates data to the target array, providing an extremely low RPO. A lower RPO means you can accomplish failover to a disaster recovery site with minimal data loss.

Pure designed ActiveDR to prioritize front-end performance, ensuring that latency-sensitive applications are not impacted by replication. ActiveDR replication is asynchronous, which means it does not require the replication target array to acknowledge application writes to the source array. Therefore, ActiveDR has no latency impact on host applications due to distance between the arrays. Because ActiveDR doesn't impose network latency constraints, you can use your existing network infrastructures with virtually no distance limitation between the arrays.

With single-command failover plus intelligent failback, ActiveDR is simple to implement, test and manage.

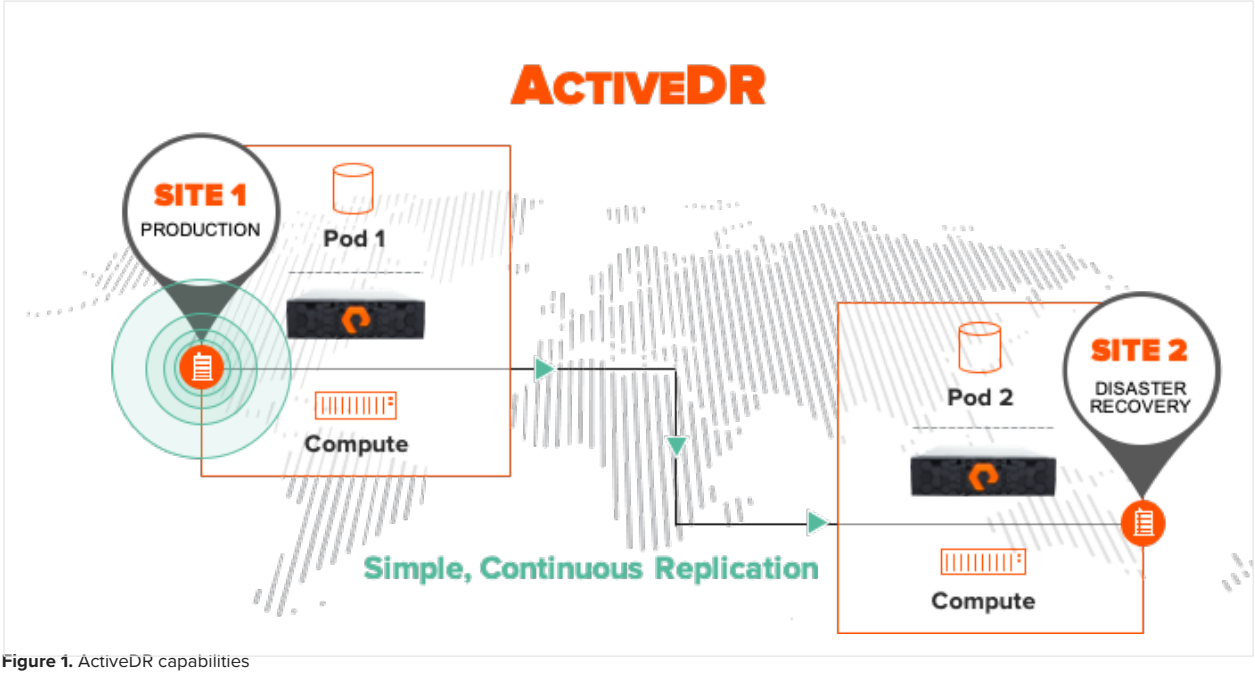


**Figure 1.** ActiveDR capabilities

## ActiveDR Delivers

- **Near Zero RPO:** Continuous streaming of writes between Pure Storage FlashArray™ systems provides a near-zero RPO without the performance impact of synchronous replication.
- **Active-Passive:** Automatically creates non-writable replica volumes at the target site that can be pre-connected to simplify DR workflows.
- **Fast Recovery / Failover Time:** Performs failover to the volumes at the target site with a single command. Failover includes both volumes and any corresponding protection group snapshots.
- **Test Failover without Compromise:** Tests failover without stopping replication, allowing the RPO to be maintained while DR testing is performed.
- **No Journals to Manage:** Continuously and automatically changes tracking, eliminating the need to provision or monitor journal devices.
- **No Latency Restrictions:** Supports replication at nearly any distance, and latency between arrays does not affect front-end application performance.
- **No Bolt-ons and No Licenses:** Simply upgrade the Pure Purity operating environment to use ActiveDR with no external hardware or costly software licenses required.

## ActiveDR Overview

ActiveDR has three core components: Storage management containers called pods, replica links, and connected Pure FlashArray systems.

- **Pods** organize storage objects and configuration settings into groups that can be failed over and back together.
- **Replica Links** provide directional and intelligent auto-reversing replication between pods. You enable ActiveDR by creating these replica links.
- **Connected FlashArray Systems** compress and continuously replicate data to enable a near-zero RPO for the contents of the pod.

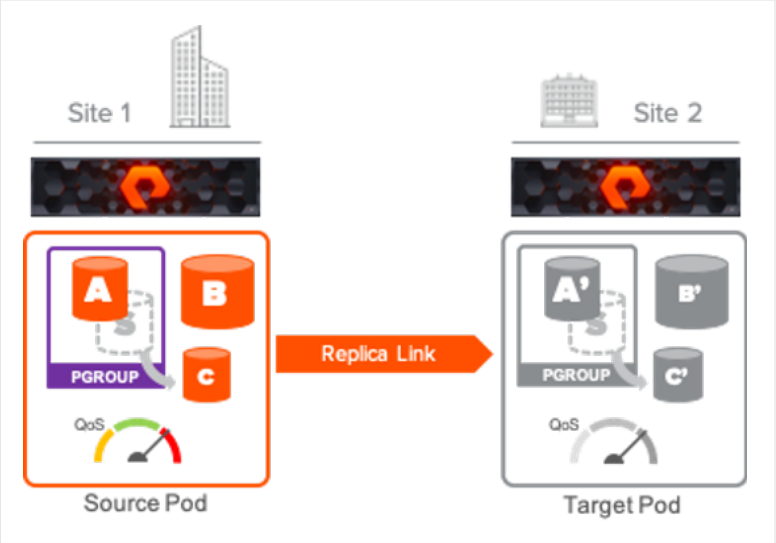


**Figure 2.** Replication links between pods

ActiveDR does more than simply replicate your data. Pods are versatile storage object containers that:

- Define which objects, and the settings associated with those objects, are replicated
- Contain volumes, volume snapshots and protection groups
- Contain and replicate configuration settings such as protection group snapshot schedules, snapshot retention policies, and QoS volume limits.

ActiveDR pod-based replication simplifies multi-site storage management by ensuring all configuration changes on the source array are synchronized to the target array. You can make configuration changes (like those listed below) inside the source pod. These changes are automatically replicated to the target with no additional steps required.

- Create new volumes in the pod.
- Resize volumes in the pod.
- Create snapshots in the pod.
- Clone volumes or snapshots to create new volumes.
- Change volume QoS limits settings.

- Configure protection groups to schedule snapshots.
- Change protection group snapshot schedules and retention policies
- Change protection group volume membership.

These capabilities make DR storage failover events extremely simple to manage. When you failover with ActiveDR, your target environment has the same configuration and object history as your production environment. You no longer need to worry about configuration drift or spend time manually replicating config changes or creating or resizing target volumes.

You can deploy ActiveDR between two FlashArray systems within a site or across multiple sites separated by distance. A multi-site deployment has the advantage of protecting the copy of data in a physically separate and potentially distant location. ActiveDR replication technology has no associated write latency impact on the primary array. That means the distance between the FlashArray systems can be nearly unlimited.

ActiveDR supports multi-direction replication for different pods. You can configure multiple pods in different directions between two FlashArray systems.

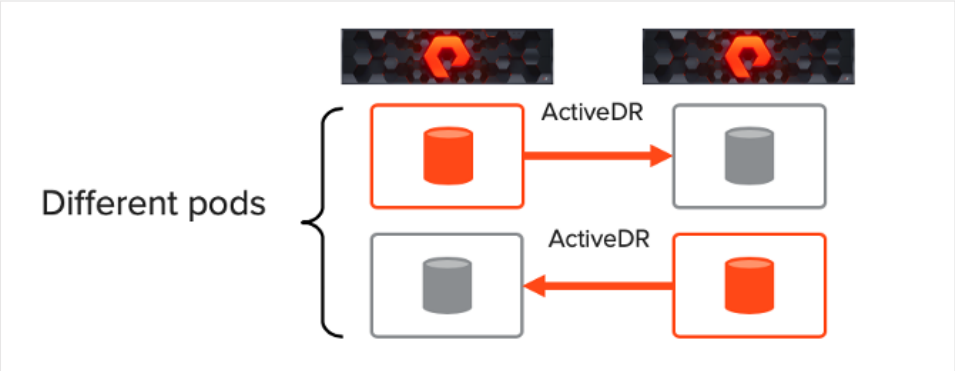


**Figure 1.** Multi-direction replication

## ActiveDR Operations

ActiveDR allows the creation of an active-passive data replication relationship between two pods in two separate FlashArray systems. This relationship is referred to as a replica link. We consider the target pod to be passive while in a demoted state, because its content, including its volumes and configuration, are write-disabled.

Although the volumes within the target pod are write-disabled, you can pre-connect them to hosts at the target FlashArray system to reduce the number of steps required during a failover. However, the primary purpose of the target pod is for receiving the data from the source rather than for reading data while in a demoted state. Various host operating systems might have different behaviors while mounting and reading filesystems on write-disabled volumes.

Pods that are continuously replicated using ActiveDR between two FlashArray systems are said to be linked. Each FlashArray system can support multiple pods. After two pods are linked and replicating, the target pod contents will include the same volumes and volume attributes (i.e., volume size, I/O limit, bandwidth limit, etc.), as well as the same protection groups and volume snapshot history as the source pod.

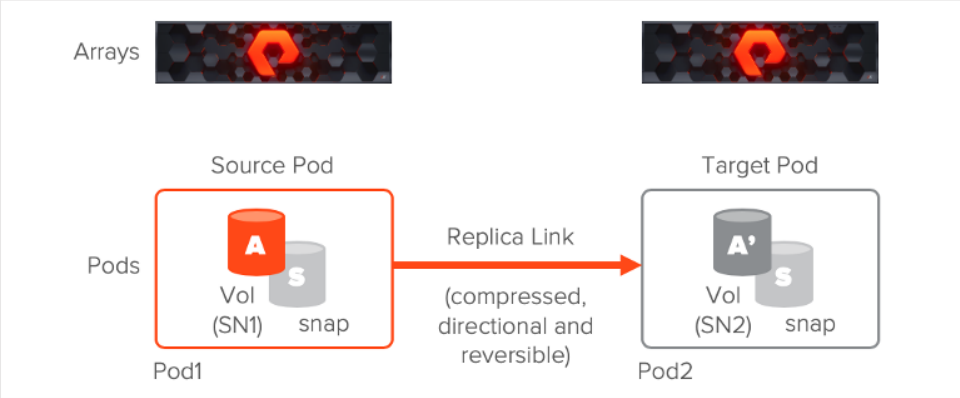


**Figure 2.** Linked pods

The content on the target array will be slightly behind the source depending on what RPO is being achieved by the ActiveDR replication engine. The RPO is reflected in the lag reported on the replica link.

You can use protection groups inside pods to schedule and manage snapshots at the source. Scheduled snapshots will automatically appear at the target when the visible contents of the target pod are periodically refreshed.

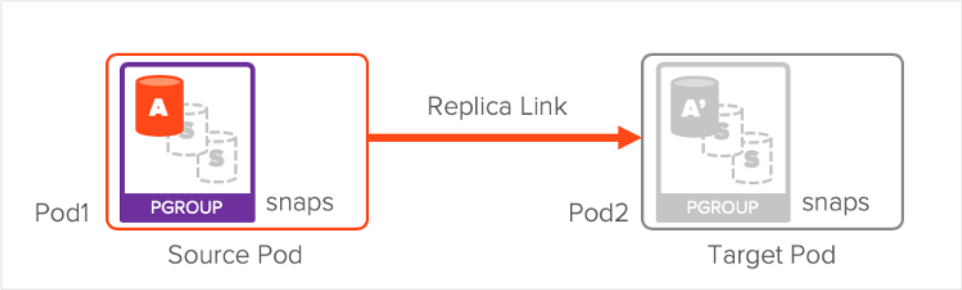


**Figure 3.** Replicating volumes

Pods also provide volume namespaces. In other words, different volumes may have the same volume name if they are in different pods. In figure 5 above, volume A in Pod1 is replicated using ActiveDR to volume A in Pod2. The full volume name on the target FlashArray system would be Pod2::A. You can configure ActiveDR in just a few easy steps:

1. Connect two FlashArray systems for replication.
2. Create a pod, create new volumes, or move existing volumes into the pod.
3. Create a replica link between a source and target pod.

After these steps are completed, ActiveDR will start the initial synchronization of data from the source to the target.

## Initial Synchronization and Baselining

After a replica link is created, ActiveDR automatically begins the initial synchronization process by using Purity's powerful and highly efficient asynchronous snapshot-based replication engine. The first synchronization of content between the replication source and target pods is referred to as baselining. The replica link status will indicate a 'baselining' status when a replica link is initially created between two pods, when the direction of replication is reversed, or when replication links have been disconnected and then restored.

## Normal Replication Operation

Once baselining completes, ActiveDR automatically transitions to its normal 'replicating' mode where low RPO continuous replication is used.

As illustrated below in figure 6, when ActiveDR writes data into the source pod (1) the source FlashArray system compresses and tracks the data and then (2) persists and tracks the write operations back to the host (3). The source FlashArray system then continuously streams (4) the compressed data to the target FlashArray, which efficiently tracks (5) the incoming changes and stores them using Purity's native deduplication and metadata technologies without requiring the additional space for a traditional journal (6).

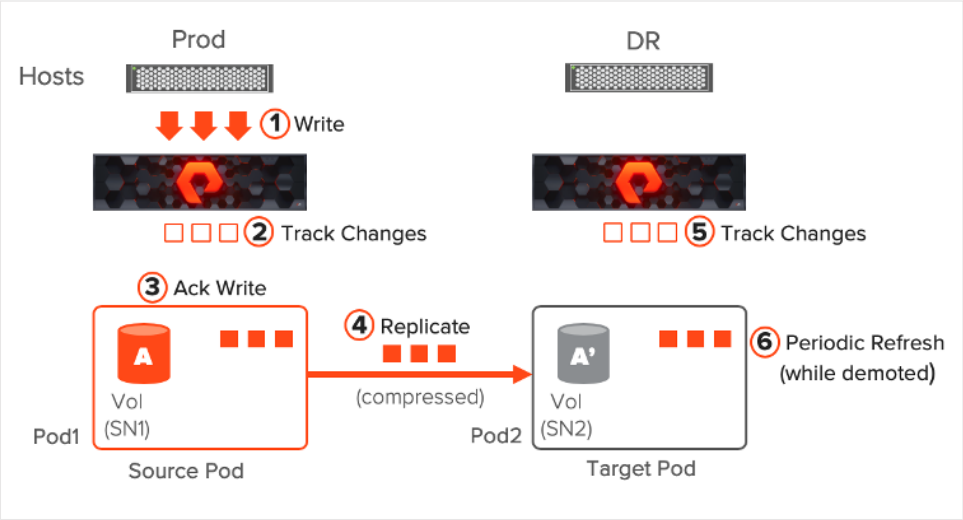


**Figure 4.** Normal replication operation

If you view the contents of the target pod while it is demoted, the visible content inside the target pod (i.e. volume properties, volume data, snapshots, protection groups, and volumes) will be periodically refreshed. When the pod is promoted for a failover or a test, it will be presented with the contents from the latest RPO available on the target.

If the incoming write rate exceeds the available network bandwidth between the arrays, or if either array becomes overloaded with front-end workloads, ActiveDR will automatically transition from the near-zero RPO engine to the async engine in the background. ActiveDR in this state will use the async engine to forward periodic updates to the target array. The RPO of the replica link will reflect time between these updates as each periodic transfer to the target is completed. Once the network or performance issue is addressed, ActiveDR automatically transitions back to the near-zero RPO engine.

Should replication be interrupted (e.g., if the arrays become disconnected from each other), ActiveDR can resynchronize the source and target pod without re-sending all the content. Purity maintains internal consistency checkpoints for the purpose of restarting replication or for resynchronizing pod content. The pod acts as the unit of consistency, ensuring that multiple volumes within the same pod remain write-order consistent.

## Failover Preparation

Volumes at the target have different volume serial numbers than the source volume. This allows you to manage the target pod volumes independently and prevents host applications and multipathing software from mistakenly treating the source and target volumes as the same volume. The serial numbers on the target volumes are different because ActiveDR is not a synchronous transparent failover replication solution. Instead, ActiveDR is meant for DR strategies typical of async replication solutions that involve a manual decision to activate the target and perform a failover. You can automate this process via scripts or APIs.

In preparation for failover, you can pre-connect hosts to the write-disabled volumes at the DR site (in the demoted pod). This allows devices and paths to be pre-discovered and pre-created on the DR hosts to make the overall failover process shorter and simpler. Though hosts can be connected to the target pod volumes, these volumes will be read-only while the target pod is in a demoted state.

If volumes are mounted while they are write-disabled, ActiveDR will display the contents of the data from the last periodic refresh. When a failover is performed, the content from the current RPO is presented in the pod.

***Note:*** *The ability to access and read the volumes while they are in a read-only state will be dependent on the host operating system's and/or application's ability to mount and read a read-only volume as well as its multipath capabilities with read-only devices. This capability can vary by application version.*

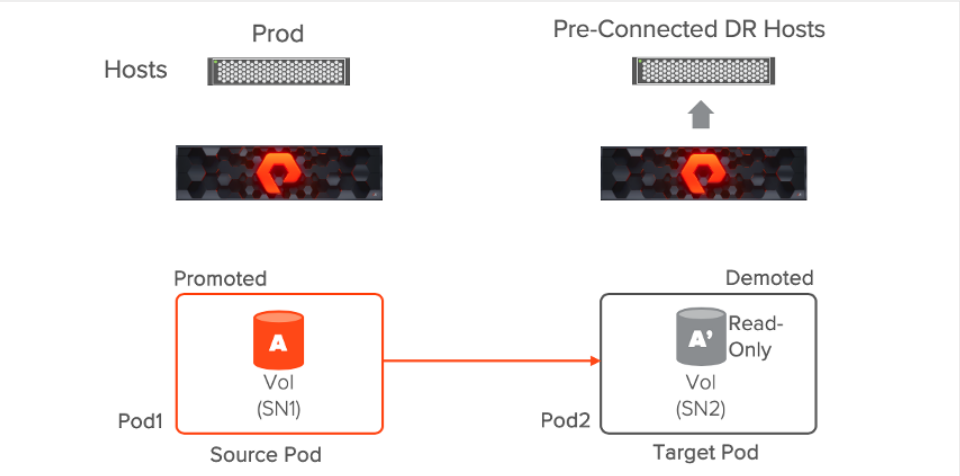


**Figure 5.** Failover preparation

When a target pod is promoted, the most current pod content received by the replication target FlashArray system is written to the target pod volumes. This makes the volumes readable and writable.

## Test Failover

Testing failover with ActiveDR is simple and efficient. At the target site, perform test failover by taking these steps:

1. Promote the target pod, wait for the status to change from promoting to promoted.
2. Unmount/mount or remount the filesystems in the hosts (if the hosts have been pre-connected).
3. Perform DR testing by starting applications or accessing the target volumes.

To perform a test failover, you must first promote the target pod (indicated by the orange outlined pod in figure 7 above). Promoting a pod takes occurs quickly. During this process, the pod's status will change from 'Demoted' to 'Promoting' and then to 'Promoted'.

Promoting the target pod makes the pod volumes writable and allows application testing to begin (indicated in purple in figure 8 below). The volume content presented to the hosts and their applications will be at the point in time contained by the last successfully completed replication transfer (indicated by the orange squares).

***IMPORTANT:*** *Before applications are started at the DR site, the DR host OS must unmount (if mounted) and then remount the file systems. This ensures that any host file system caching, I/O buffering, or application caches are cleared and do not contain stale or invalid data from previous target pod volume content. Local filesystems could have different memory states that are inconsistent with the data on the target pod, because ActiveDR modifies the contents of the target directly without notifying any locally connected hosts.*

While the target pod is promoted, and if the arrays remain connected, ActiveDR will maintain replication from the source in the background by continuing to stream new content (indicated in green) to the target array.

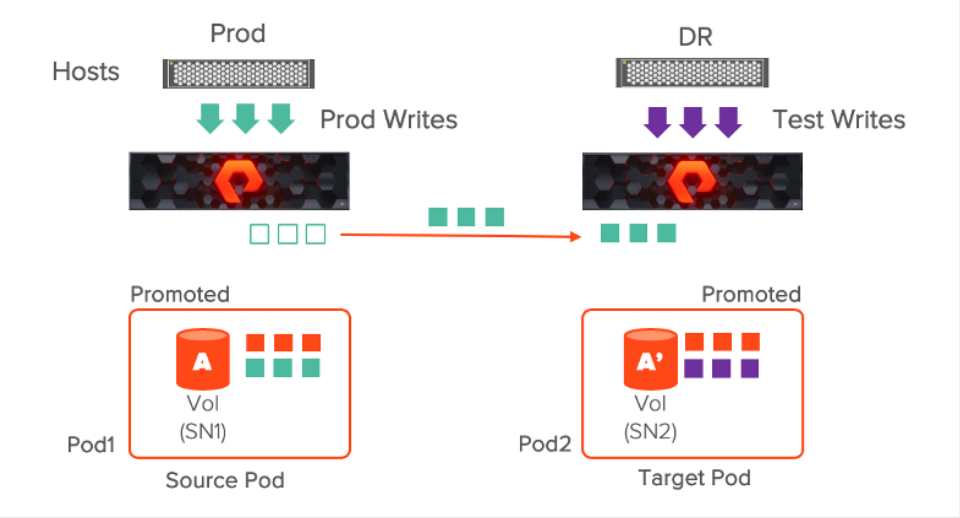


**Figure 6.** Test failover

After promoting the pod and remounting the filesystems (if the hosts were pre-connected), you may perform DR testing by writing into the pod (shown in figure 8 above in solid purple squares).

The replicated content will not be shown in the target pod while the pod is in a promoted state. The target FlashArray system will store the replicated content in a separate accounting bucket using metadata and pointers for space efficiency (shown in figure 8 above in solid green squares). The capacity required for this is marginal and depends on how much change rate there is and how long the pod remains promoted while replication is happening. It essentially has the same capacity costs as maintaining snapshots for the same period of time. This capacity is displayed in the UI as 'Replication Space'.

When failover testing is complete, simply demote the target pod. Demoting a target pod will cause any test data written into that pod to be discarded from the pod. The content that was replicated from the source will then be attributed to the target pod.

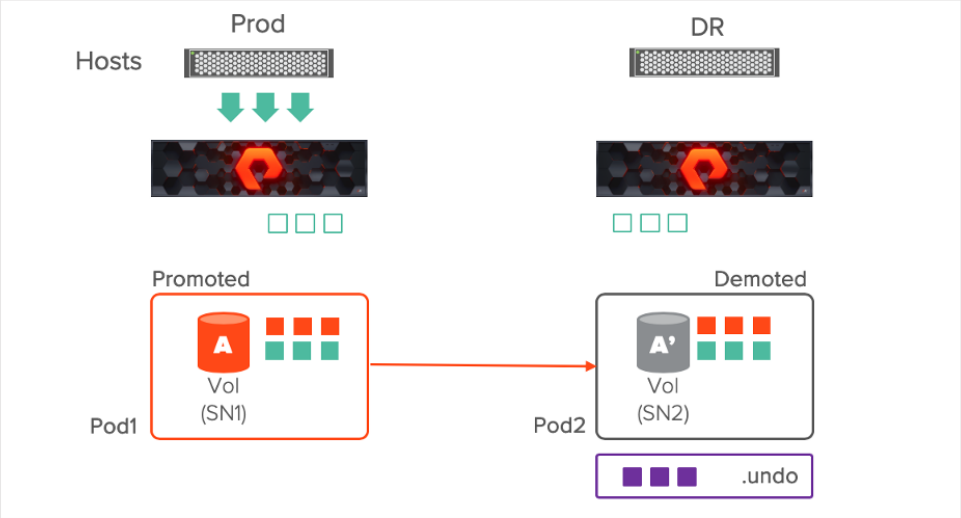


**Figure 7.** Creating an undo pod

ActiveDR simplifies DR testing and does not require that replication be stopped. This enables more frequent DR testing. This can mitigate risks associated with failover events by increasing the confidence in DR plans and processes.

***Note:*** *Any changes made to the content of the target pod while it was promoted (shown in solid purple squares in figure 9 above) are recoverable after being demoted. When a pod is demoted, Purity creates an undo pod. This undo pod is a special object that, if cloned, can be accessed as a new pod to gain access to that data. Undo pods are automatically eradicated according to Purity's default eradication time.*

## Real Failover

The real failover workflow follows the same initial steps as the test failover and uses the same promote command to get things started. To perform real failover:

1. Promote the target pod, wait for the status to change from promoting to promoted.
2. Unmount/mount or remount the filesystems in the hosts (if the hosts have been pre-connected).
3. Start applications.

***Note:*** *Typically performing DR testing without stopping replication requires that you use different commands or APIs for test failover versus real failover. For example other technologies may require you to create a clone to do a test without stopping replication. This presents risk that functions may not work during real DR failovers if they are not regularly tested. ActiveDR uses the same command for both test failover and real failover, ensuring that any runbook or orchestration steps are the same during a test or in an actual failover event.*

Following the completion of the target pod promotion, refresh any host file systems mounted on the promoted target pod volumes (unmount/remount from the host) and start up applications at the target site.

With any non-synchronous replication technology, it's fairly typical during a real failover to have some writes to the source pod that don't get replicated to the target (shown in green in figure 10 below). Another common situation is when replication links are lost, preventing data from being replicated from the source pod to the target pod. This may be due to the source FlashArray system being offline and/or disconnected from the target FlashArray (shown in grey in figure 10 below).

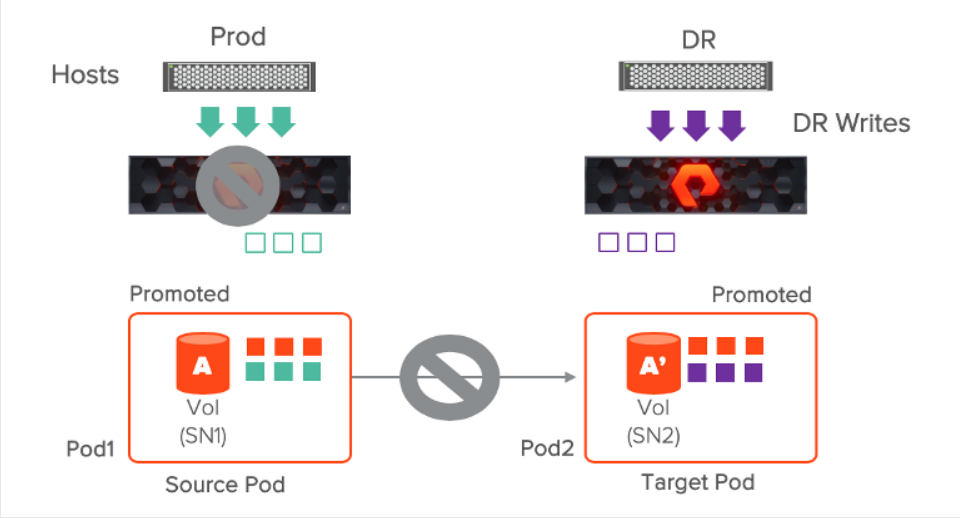


**Figure 8.** Writes not replicated during real failover

If both FlashArrays remain online and connected, then replication continues in the background, just as it does during a test failover scenario. If a target pod is promoted, the replication will not be disrupted, nor will the target pod volumes be overwritten. This is accomplished by the target array's ability to track incoming replication data in the background, just as it does during a test failover scenario. Promoting a target pod means any incoming replication data is only tracked in the background, as it does during a test failover. If replication continues following a site failure and restoration of power, the administrator can suspend ActiveDR replication from the original source FlashArray system by pausing the replica link.

## Reversing Replication for Re-Protection

After a real failover, when the original source FlashArray has returned to service, the ability to replicate data back to the restored FlashArray is often required. To accomplish this:

1. Stop the applications at the original source.
2. Demote the original source pod.

***Note**: When the original source site is recovered, you must stop any applications located there before reversing the replication back to the original source pod.*

To reverse replication, simply demote the original source pod on the original source FlashArray system by selecting the **skip quiesce** option. This will make the original source pod volumes read-only and will reverse the direction of replication. The **--quiesce** option is used for planned failovers as discussed later in this document.

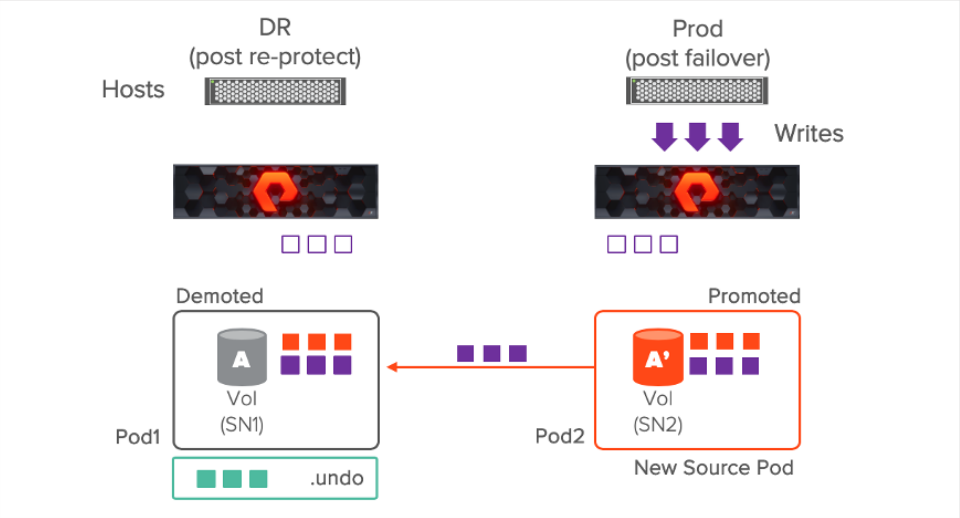


**Figure 9.** Automatic replication reverse

Whenever you demote a pod that is the source of a replica link, if the other pod is promoted, ActiveDR will automatically reverse replication, ensuring that the new application data at the other site is protected as quickly as possible. This eliminates risks associated with manual processes and automates the reversal of replication relationships. Automatic replication reversal does not happen during DR tests because the DR test pod is the target, not source, of the replica link.

Demoting the original source pod will also save a temporary copy of the source pod content (prior to replication reversal) in the recycle bin for 24 hours (shown in green in figure 11 above). If needed, you can clone the .undo pod so that you can recover any data written but not replicated before the outage.

## Planned Failover with ActiveDR

You can use ActiveDR to perform planned failovers or migrations between sites with a short interruption for cutover.

The planned failover process assumes you desire a controlled failover and can gracefully shut down applications before the ActiveDR failover is performed and the applications are restarted at the target site.

1. Stop the applications at the source.
2. Demote the source pod using the quiesce option.
3. Promote the target pod.
4. Start the applications at the target.

The first step in a planned failover is to stop all applications that are writing to the source pod volumes. Next, demote the replication source pod and select the quiesce option. The quiesce option tells ActiveDR to put the replica link into an idle state after all content has been sent to the target. Quiescing takes just a few moments.

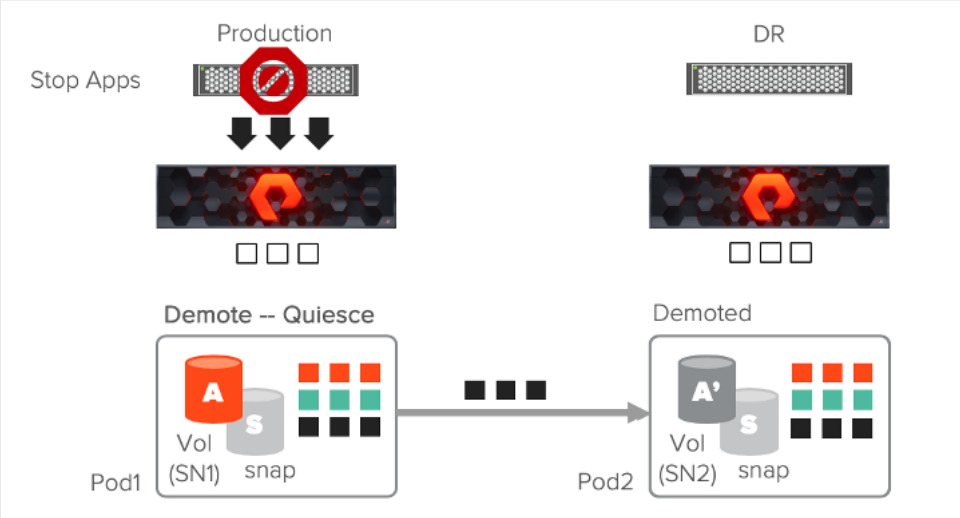


**Figure 11.** Planned failover

The last step is to promote the target pod (just as you would for any failover) and then refresh the host mounts as described above. At that point, you can start the failed-over production applications on the target pod.

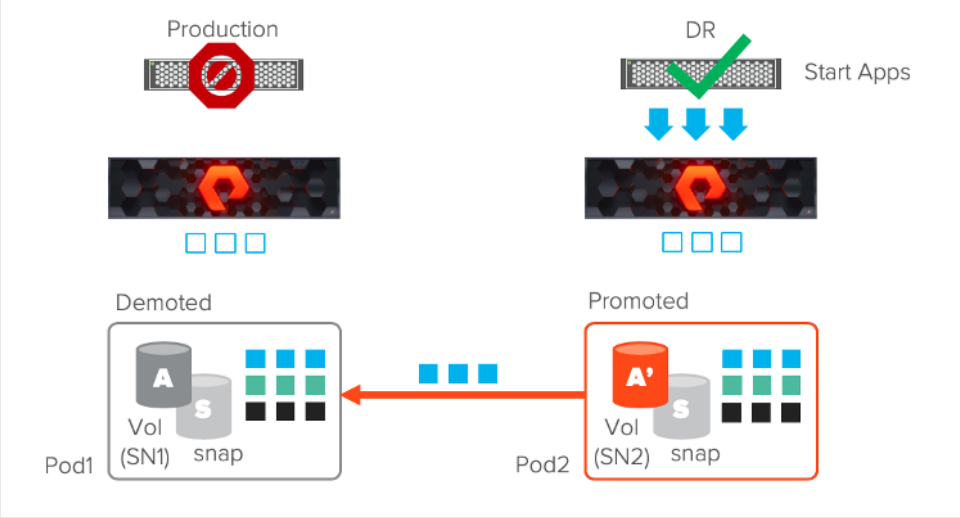


**Figure 10.** Automatic replication reversal

ActiveDR will automatically reverse the direction of replication when the target pod is promoted if the original source pod has already been demoted with the quiesce option. As in the failback after failover, automatic replication reversal eliminates any risk associated with manual steps or processes.

As mentioned, quiescing takes just a few moments. In the rare event that a replication outage occurs during quiescing. ActiveDR provides an option on the target to abort the quiesce. Aborting the quiesce will promote the target pod to the latest consistent RPO that was replicated to the target.

# Application and Host Failover with ActiveDR

## Data Consistency

Data consistency and the type of data consistency provided by ActiveDR are important concepts that impact host failover processes.

ActiveDR replication provides crash-consistent or point-in-time consistency for promoted target pod volumes. An internal checkpoint mechanism captures the state of a possibly changing pod at an instant of time in which no updates are in "mid-flight." This point reflects all writes to the pod prior to the checkpoint in their entirety. No part of any update that occurred after a checkpoint is reflected in the image of data replicated to the demoted target pod volumes.

Crash consistency of a pod does not imply consistency at the application level. In the case of a database, for example, the database management system may be updating a database block while a user-managed backup is reading it. If a crash occurs at such an instant, a copy of the block may be internally inconsistent—its header may indicate that contents have been updated, but the updates are not present in the block content. Database blocks in this state are said to be fractured.

While crash-consistent pods may not always be consistent from the application or database point of view, they can be recovered to a consistent state because databases and many other applications carefully order writes to persistent storage. For example, they may record their intention to update in a log file or via a journaled file system before writing other data. As long as target pods preserve the order of writes (updates), the application or database can recover to a consistent state using the target pods and the crash-consistent volumes they contain.

## Application and Host Failover

ActiveDR replication is asynchronous and provides crash-consistent copies of volumes, protection groups, and snapshot history in a target pod on a second FlashArray system. Because the replication is not synchronous, the target pod contains time-lagged data (as determined by the RPO) and its volumes have different volume serial numbers than the corresponding source volumes.

ActiveDR does not automatically or transparently failover to the target FlashArray upon loss or unavailability of the source. The ActiveDR failover process for applications and hosts must be triggered by an administrator. While ActiveDR failover is not automatic, you can automate it with an API-based script or a purpose-built, third-party DR automation tool.

Because of the nature of asynchronous replication, storage administrators typically make a conscious decision to perform a failover in the event of a disaster. It's important to consider all aspects of the failover process, including the degree of automation used to start up the DR site, the process for redirecting users to the DR site, any side effects associated with the loss of data on failover (as determined by the RPO), the expected length of the outage, and more. In some cases, the storage administrator may decide to wait for the outage to be resolved rather than engaging a full failover. In any case, ActiveDR provides the most seamless and simple failover process for the storage administrator.

This section will provide an overview of the host and application failover process:

1. Disaster is determined or declared.
2. Promote demoted target pod(s).
   ***Note:*** *If a target pod is already promoted and the latest copy of data is required for failover, the target pod will need to be demoted and then promoted to be refreshed with the latest copy of data.*
3. Unmount all currently mounted file systems on target pod volumes on DR hosts.
4. Re-mount all unmounted file systems target pod volumes on DR hosts.
5. [If required] Modify DR application startup and configuration files to accommodate any new network IP addresses, new host paths to volumes, and new file system mount points.
6. Confirm all target pod volumes required for DR applications to start are accessible to DR hosts.
7. Start DR applications (with application specific recovery checks as appropriate).
8. [Optional] Perform a planned migration back to the original source FlashArray system using the planned failover process described earlier.

## ActiveDR Solution Requirements

### Performance Requirements

Continuous replication is a background process and should generally not impact the front-end host I/O performance of the FlashArray system.

Achieving or maintaining a near-zero RPO (an RPO of a few seconds or less) depends on several factors. The primary factors in determining the achieved RPO are:

- The front-end host workload on the source and target FlashArray systems.
- The replication network bandwidth between the source and target FlashArray systems
- Incoming write rate to the source pod volumes

If the incoming write rate exceeds the available replication network bandwidth between the FlashArray systems for an extended period of time, or if either the source or target FlashArray systems become overloaded and cannot maintain replication throughput as well as front-end workload, ActiveDR will not be able to maintain a constant near-zero RPO. If the solution cannot sustain a near-zero RPO due to exceeding any of the above factors, ActiveDR will automatically transition into asynchronous periodic replication mode. This periodic replication will provide a best-effort async RPO depending on the workload change rate. The periodic RPO is not configurable.

## Replication Network Requirements

The replication network enables you to make the initial transfer of data to the target pod, to continuously stream data and configuration information between the arrays, and to resynchronize a pod.

### General Network Requirements

- 4x 10GbE replication ports per array (two per controller). Two replication ports per controller are required to ensure redundant access from the primary controller to the other array.
- 4x dedicated replication IP addresses per array
- A redundant switched replication network. Direct connecting Pure Storage FlashArray systems for replication is not possible.
- Adequate bandwidth between arrays to support continuous ActiveDR replication as well as bandwidth for initial synchronization and for resynchronizing. Required replication bandwidth depends on the write rate of the hosts at both sites.

### Bandwidth Requirements

The bandwidth required to operate ActiveDR is dependent on the incoming write rate from the application at the source FlashArray system. The FlashArray will maintain compression on the wire between the arrays, reducing the overall bandwidth required to make replication as efficient as possible. An additional 30% replication network bandwidth above the incoming write rate should be available to support resynchronizing pods that have disconnected and reconnected and to accommodate unexpected write bursts.

### Latency Requirements

There are no maximum latency limits between FlashArray systems using ActiveDR. This is because the latency between arrays does not affect host-side latency as it does in synchronous replication solutions. However, higher latency networks will tend to provide less bandwidth and will translate into inability to maintain a near-zero RPO if the incoming write rate exceeds the available replication network bandwidth.

To sustain a near-zero RPO, you need to maintain adequate replication bandwidth to support continuous replication of incoming writes from the host to the target array as described above.

### Replication Network Connectivity

Replication connectivity for ActiveDR requires two Ethernet ports per controller. They must be connected via a switched infrastructure such that every replication port on one array is able to connect to every replication port on the other array.

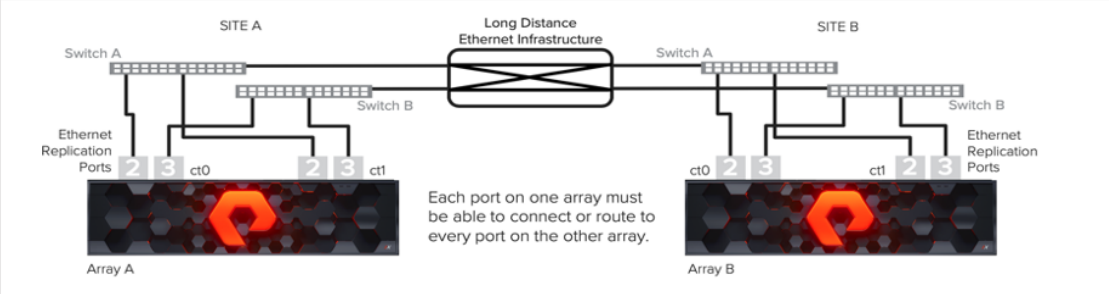


**Figure 12.** Replication network connectivity

This allows services to move from port to port depending on local failover events in each array. For redundant configurations using dual switches, each controller must have a connection to each local switch. In addition, the switching infrastructure must still allow all replication ports to be able to connect to each other.



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