Oracle Linux

Backing up Files and Storage Volumes for Disaster Recovery

F38619-11
November 2022

Oracle Linux Backing up Files and Storage Volumes for Disaster Recovery,

F38619-11

Contents

Preface

Oracle Linux: Backing Up Files and Storage Volumes for Disaster Recovery describes how to configure your Oracle Linux system to automatically back up and restore files, folders, and storage volumes.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at https://www.oracle.com/corporate/accessibility/.

For information about the accessibility of the Oracle Help Center, see the Oracle Accessibility Conformance Report at https://www.oracle.com/corporate/accessibility/templates/t2-11535.html.

Access to Oracle Support for Accessibility

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit https://www.oracle.com/corporate/accessibility/learning-support.html#support-tab.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1 About Backup and Disaster Recovery

A backup and disaster recovery strategy is a key component in a comprehensive operational maintenance plan. Your disaster recovery strategy should aim to enable the continuation or recovery of systems, infrastructure, and data following a destructive disaster event.

Disaster events can be natural events, physical hardware failures, software bugs, or human induced data destruction through error or malice. In all cases, you should have a plan that caters to recovering critical systems and data with as little downtime as possible and minimal data loss.

The following are different disaster events that may require different procedures and policies:

- **Data center failure**

  A disaster recovery strategy that caters to data center failure can be costly but can also effectively mirror data and systems in several different geographical locations. This approach can help to recover from an event that can affect an entire data center at a particular geographical location. To handle these types of events, physical systems must be available in multiple locations and data must be replicated to each location so that systems can be restored quickly.

  The cost of a disaster strategy that caters to this type of failure can be significantly reduced by using cloud-type services, such as Oracle Cloud Infrastructure, that provide services across geographical locations.

  Oracle Linux additionally provides geographical data-replication facilities in tools such as Gluster Storage for Oracle Linux. See Working With Data Mirroring for more information.

- **System failure**

  A system failure strategy should cater to providing physical hardware to replace components or whole systems in the event that hardware fails or a destructive action removes full system functionality. In some cases, physical hardware may address component failure by providing redundant components; but, your strategy should take into account the possibility of total system failure. You should plan around how quickly you are able to physically replace a complete system for each business-critical resource that you may need to restore. Ideally, this strategy should also include a plan to deploy software configuration information and data, as required.

  Oracle Cloud Infrastructure can help reduce the total cost of ownership when planning for system failure, as it provides the facilities to create custom system images and configuration entries based on existing infrastructure for quick deployment of a new system or configuration, as needed.

  You can achieve further system resilience through virtualization or container solutions, which help abstract system processes and functionality from the physical hardware. Use of virtualization or container services also provides the opportunity to create images or deployment plans to rapidly recover services, as needed. See Oracle Linux: Podman User's Guide and Oracle Linux: Oracle Container Runtime for Docker User's Guide for more information about container services. See Oracle Linux: KVM User's Guide for information about virtualization in Oracle Linux.

- **Disk or volume failure**

Typically, the most frequently catered-for event in every disaster recovery strategy is disk or volume failure. It is not uncommon for disk failures to occur; although, the frequency with which these events happen is significantly reduced as hardware evolves. Backup and mirroring software are low-cost, usually easy to implement on Oracle Linux, and can help with the mitigation of these issues.

Disk and volume failure are usually handled by performing some kind of data mirroring or replication. Data resilience is often achieved through disk redundancy by using RAID-1 mirroring, volume snapshotting, and traditional backup methods. See Working With Data Mirroring for more information.

Volume-level snapshotting can help to replicate data across volumes and is discussed in more depth in Working With File System Snapshots.

Full data backup, which is described in Managing Backups With ReaR, also provides some level of platform recovery in the case of system-level failure.

Note that when using Oracle Cloud Infrastructure, block devices that function as disks within your instances have built-in data replication capability across multiple servers to guarantee availability and uptime. By using Oracle Cloud Infrastructure and storing data on dedicated block volumes, mitigation against disk or volume failure is done automatically.

- **User and software events**

  User and software events can include malicious attacks on systems or inadvertent errors that result in the destruction or corruption of data on a file system, as well as software bugs or updates that may result in unintended configuration changes and other data corruption. Rapid rollback to a known, working environment is highly desirable in any disaster recovery strategy.

  Traditionally, this domain has largely been handled by full and regular data backups. This approach is still useful, but recovery can be slow and usually requires some downtime. Nonetheless, this approach should be used in combination with other, quicker solutions to maximize protection. See Managing Backups With ReaR for more information about managing backups.

  The file system snapshotting feature provided by Btrfs can reduce the amount of time that is required to return a system to a known working state. For more information and instructions, see Working With File System Snapshots.

A comprehensive disaster recovery plan should utilize a combination of the tools that are suited to your platform, environment, and hosting needs.

Cloud-based services typically provide the tools and built-in redundancy to assist with mitigation against data loss, as well as speed up recovery time. However, even in these environments, you may consider using a combination of additional tools and facilities to cover all potential disaster scenarios. For example, by using file system snapshotting on a cloud instance, you are able to fine tune system rollback even after a basic software update.

For systems that are located within your specific data center, you are likely to use a wider range of tools and services to build in resilience and durability.

This document provides pointers to the different tools that are available in Oracle Linux for achieving a more comprehensive disaster recovery strategy by using software that is native to the operating system. In addition, more thorough coverage is also provided for the Relax-and-Recover (ReaR) and data backup tools that are provided with Oracle Linux.

2

Managing Backups With File System Snapshots and Data Mirroring

Working With File System Snapshots

You can configure supported file systems to use Copy-on-Write (CoW) functionality to replicate data between a snapshot and a volume or subvolume. Snapshots are an inexpensive use of disk space and an efficient way to roll back small changes.

File system snapshots should be used as part of a broader back up and recovery strategy. Snapshots cannot protect against hardware failures, but they can provide a rapid recovery mechanism in the event of a software failure or user error.

You can automate the creation of snapshots with the `snapper` utility.

Managing Snapshots With Btrfs

On Btrfs, you can create snapshots in any directory within the file system, and the file system itself will monitor and maintain the consistency of each snapshot. This monitoring and maintenance makes Btrfs snapshots incredibly reliable and provide significant performance gains over using snapshot functionality provided by the Logical Volume Manager (LVM).

The `snapper` utility that is provided on Oracle Linux to assist with file system snapshot creation and management can help to simplify and automate regular timed snapshots so that it is easy to roll the system back to a particular time. The utility also comes with a plugin for `yum` or `dnf` so that it is able to automatically generate a snapshot immediately before and immediately after any software updates, installations or removal. Snapshots that are tied to software changes simplify any recovery from inadvertent configuration changes, conflicts, incompatibilities and a number of other similar disaster scenarios.

For more information about managing Btrfs formatted volumes, see Oracle Linux 8: Managing Local File Systems and Oracle Linux 9: Managing Local File Systems.

Managing Snapshots With LVM

Logical Volume Manager (LVM) provides volume abstraction for the operating system so that logical volumes can span multiple disks. As a result, LVM helps to facilitate continuity of service while allowing for hard-drive replacement, partition resizing and volume backup. LVM allows for the creation of snapshots of entire volumes.

When you create snapshots with LVM, the underlying file system is unaware of snapshots you have created, because the snapshot is created at the volume level. LVM snapshots maintain a mirror of the logical volume at a specific point in time, but file system consistency is not always guaranteed. You must mount the snapshot volume after it is created if you need access to it from within the operating system.

Although LVM snapshots only track changes to the volume and are therefore efficient in terms of storage space, the implementation of Copy-on-Write can impact general system performance negatively when snapshots are left in place because the snapshot effectively tracks and stores metadata for every change happening on the original volume. For this reason you have to plan your snapshot volumes carefully beforehand to ensure that they have enough space to track these changes and you should regularly clean up snapshots after you are finished using them. LVM snapshots are primarily designed to provide a static and unchanging image of a volume at a point in time to facilitate stable backup, but can also be used to rollback changes where there are planned maintenance windows.

For more information about managing storage with LVM, see Oracle Linux 8: Managing Storage Devices and Oracle Linux 9: Managing Storage Devices.

Working With Data Mirroring

You can ensure resilience against data loss and hardware failures by replicating data in multiple places. By mirroring data in real time, you can also preserve the accuracy and integrity of that data when you perform a disaster recovery operation.

Managing Geo-Replicated Data With Gluster Storage for Oracle Linux

Gluster Storage for Oracle Linux is distributed and replicated storage that provides durability by virtue of its ability to replicate data across multiple systems synchronously.

This service additionally provides the ability to asynchronously mirror data across different geographically located systems through its geo-replication features. Gluster Storage for Oracle Linux can help your enterprise recover in the case of full data center failure.

For more information about how to mirror data across locations for disaster recovery purposes by using geo-replication, see Oracle Linux: Gluster Storage for Oracle Linux User's Guide.

Managing Data Mirroring Hosted on Oracle Cloud Infrastructure

Block volumes that are hosted on Oracle Cloud Infrastructure have built in data resilience and durability. All volumes are automatically replicated and stored redundantly across multiple storage servers with built-in repair mechanisms. For more information, see Overview of Block Volumes.

On some compute instances on Oracle Cloud Infrastructure you are able to use locally attached NVMe devices for extremely low latency and high performance block storage. In that case where you have locally attached NVMe devices, this storage is not protected with the same data resilience and durability for general block storage provided on Oracle Cloud Infrastructure. To mitigate against hardware failure, you should consider setting up software RAID to mirror data across multiple devices. For more information see Protecting Data on NVMe Devices. For more general information about configuring RAID on Oracle Linux systems, see Oracle Linux 8: Managing Storage Devices and Oracle Linux 9: Managing Storage Devices.

Managing Block Device Redundancy With Software RAID

To replicate data across volumes, you can configure Oracle Linux to use Software RAID in a RAID-1 "mirror" configuration. RAID-1 is most useful for localized disaster recovery when used in conjunction with file system or volume snapshot functionality. It is less flexible and resilient than geo-replication and the mirroring that is provided by Oracle Cloud Infrastructure, but it provides an immediate on-site service that can reduce downtime in case of disk failure.

For more information about configuring RAID, see Oracle Linux 8: Managing Storage Devices and Oracle Linux 9: Managing Storage Devices.

3 Managing Backups With ReaR

ReaR is a disaster recovery tool that you can use in your local data center on Oracle Linux systems. ReaR automatically generates a bootable recovery environment, as well as external file backups.

Scheduling ReaR with Crontab requires very little ongoing maintenance and you can use ReaR to restore lost user and system files to the original directory locations.

To find out more about ReaR, visit https://relax-and-recover.org/ or see the `rear(8)` manual page.

Installing ReaR

Install the `rear` package from the Oracle Linux yum server:

```
sudo dnf install rear
```

Creating a ReaR Rescue System

Use ReaR to create a bootable device that can subsequently be used to restore the underlying system and copy data from an external source.

Before proceeding, install the `genisoimage` and `syslinux` packages, as they are both required to create a bootable recovery environment as an ISO image file:

```
sudo dnf install genisoimage syslinux
```

Configuring a ReaR Rescue System

The rescue system configuration is stored in the `/etc/rear/local.conf` file.

> **Note:**
>
> For a guide to standard configuration of all available parameters that can be used in `/etc/rear/local.conf`, refer to the `/usr/share/rear/conf/default.conf` file.

Define the storage medium by setting the `OUTPUT` parameter and the ISO image file location, which is specified by the `OUTPUT_URL` parameter:

```
OUTPUT=ISO
OUTPUT_URL=file:///mnt/rescue_system/
```

This configuration generates an ISO image file in the `/var/lib/rear/output/` directory as `/mnt/rescue_system/`*`host_name`*`/rear-localhost.iso`. You can also replace `file:///` with `nfs://` for network storage mounts.

If disk space is limited, optionally configure ReaR to omit /var/lib/rear/output/ to only generate a single ISO image file in the /mnt/rescue_system directory:

```
OUTPUT=ISO
BACKUP=NETFS
OUTPUT_URL=null
BACKUP_URL=iso:///backup
ISO_DIR=/mnt/rescue_system
```

To automatically generate a file backup each time the recovery system is generated, see Using ReaR to Back Up Files.

Generating a ReaR Rescue System

After you have configured the base settings for your rescue system, generate an ISO image file by using the mkrescue command:

```
sudo rear mkrescue
```

For information about how to create USB installation media from an ISO file by using the dd command, see Oracle Linux 8: Installing Oracle Linux.

Scheduling the Creation of ReaR Rescue Systems

Similar to most command-line tools, rear can be scripted to run automatically by using the crontab utility.

For example, you would schedule /etc/crontab to automatically generate a new rescue system every weekday at 10pm as follows:

```
0 22 * * 1-5 root /usr/sbin/rear mkrescue
```

For more information about crontab, see Use the Crontab Utility to Schedule Tasks on Oracle Linux.

Using ReaR to Back Up Files

In addition to creating a rescue system, you can optionally configure ReaR to generate a full file backup at the same time.

Creating Tarball Backups

To create a full file backup and store the results as a tarball, edit the /etc/rear/local.conf file.

1. Add values for the BACKUP and BACKUP_URL settings to output generated tar files in the /srv/backup directory:

   ```
   OUTPUT=ISO
   OUTPUT_URL=file:///mnt/rescue_system/
   BACKUP=NETFS
   BACKUP_URL=file:///srv/backup/
   ```

2. To preserve previous backups, add the NETFS_KEEP_OLD_BACKUP_COPY setting:

   ```
   NETFS_KEEP_OLD_BACKUP_COPY=y
   ```

To save disk space with those previous backups by making them incremental, add `BACKUP_TYPE` as follows:

```
BACKUP_TYPE=incremental
```

3. To set the file backup to run weekly, set the value of `FULLBACKUPDAY` to the first three letters of your chosen day. For example, to run the backup each Sunday:

   ```
   FULLBACKUPDAY=(Sun)
   ```

4. To create a full ISO image file, change the `BACKUP_URL` as follows:

   ```
   BACKUP_URL=iso:///backup/
   ```

Generating Backups With ReaR

To verify whether the file system has changed since you last generated a backup:

```
sudo rear checklayout
```

Provided that your `BACKUP` setting in `/etc/rear/local.conf` is `NETFS`, you can create three different kinds of backup with ReaR:

- Create a rescue system without a file backup:

  ```
  sudo rear mkrescue
  ```

- Create a file backup without a rescue system:

  ```
  sudo rear mkbackuponly
  ```

- Create a rescue system and a file backup:

  ```
  sudo rear mkbackup
  ```

Testing Your ReaR Rescue System

You must periodically test that you are able to restore from backups to be confident that system recovery is possible. Provision a test system on which you can safely perform a recovery without losing important data, and then follow these steps:

1. Boot the test system from recovery media for the rescue system that you generated in Creating a ReaR Rescue System.
2. Follow the instructions in Recovering a System With ReaR on the test system. For more diagnostic information, add the `-v` parameter to the recovery command:

   ```
   rear -v recover
   ```

3. If the recovery does not run correctly on the test system, revise your ReaR configuration and regenerate the rescue system.
4. Repeat these steps on the test system until the recovery process succeeds.

Recovering a System With ReaR

If your Oracle Linux installation has been rendered unbootable, you can recover your system and restore data by using a dedicated rescue environment that you have previously generated for that device with ReaR by following these steps:

1. Boot your device with the rescue system that you generated in Creating a ReaR Rescue System.
2. Select the Recover localhost option from the boot loader menu.
3. Log in to the rescue system as the `root` user. The shell environment uses the credentials that are stored in the `/root/.ssh/authorized_keys` file, if available. Or, you can manually set the `SSH_ROOT_PASSWORD` environment variable.
4. To launch the automated recovery process, run the ReaR `recovery` command:

   ```
   rear recover
   ```

5. The file structure for your system should be replicated in the `/mnt/local` directory so that you can optionally use it as the output directory for file recovery from an external source.

   For example, you could use the `scp` command to transfer a tarball over an SSH connection:

   ```
   scp root@example.oracle.com:/mnt/backups/backup.tar.gz backup.tar.gz
   ```

   Alternatively, use the `rsync` command to fetch files and directories from a network share:

   ```
   rsync -avzh root@example.oracle.com:/mnt/backups/backup.tar.gz .
   ```

   When you have successfully transferred the tarball backup that was previously generated by `rear`, extract the contents to `/mnt/local`:

   ```
   tar -xf backup.tar.gz -C /mnt/local
   ```

6. Set SELinux to relabel on the next boot by creating a blank file called `.autorelabel` in the `/mnt/local` directory:

   ```
   touch /mnt/local/.autorelabel
   ```

7. Exit the recovery environment by rebooting the system:

   ```
   reboot
   ```

   When SELinux has relabelled the entire file system, the recovered host system is bootable.

Creating Multiple Backups With ReaR

You can use different `rear` commands to create a rescue system with files from the underlying system, then back up the data from both the `/home` and `/opt` directories.

1. Common settings for `rear` are defined in the `/etc/rear/local.conf` file, but you can override individual values with separate configuration files for each scenario, for example:

   - `/etc/rear/basic_system.conf`
   - `/etc/rear/home_backup.conf`
   - `/etc/rear/opt_backup.conf`

   In these configuration files, select which folders to include in your backup and define the output tarball name. To include wider matches, you can provide wildcards (`*`) in path names that you specify.

For example, you would specify that only the /home directory is included in the /etc/rear/home_backup.conf file as follows:

```
BACKUP_ONLY_INCLUDE="yes"
BACKUP_PROG_INCLUDE=( '/home/*' )
BACKUP_PROG_ARCHIVE="backup-${this_file_name%.*}"
```

For more information and additional sample configurations, see https://relax-and-recover.org/documentation/.

2. Generate each backup by using the rear command, along with the -C option, to specify your choice of configuration file:

```
sudo rear -C basic_system mkbackup
sudo rear -C home_backup mkbackuponly
sudo rear -C opt_backup mkbackuponly
```

3. From within the rescue environment, use the same labels to recover your system, for example:

```
rear -C basic_system recover
rear -C home_backup restoreonly
rear -C opt_backup restoreonly
```

To find out more about rear command parameters and recovery options, see the rear(8) manual page.