Disk cloning

From ArchWiki

Disk cloning is the process of making an image of a partition or of an entire hard drive. This can be useful for copying the drive to other computers and for backup and recovery purposes.

Contents

Using dd

The *dd* command is a simple, yet versatile and powerful tool. It can be used to copy from source to destination, block-by-block, regardless of their filesystem types or operating systems. A convenient method is to use *dd* from a live environment, as in a Live CD.

Warning: As with any command of this type, you should be very cautious when using it; it can destroy data. Remember the order of input file (`if=`) and output file (`of=`) and do not reverse them! Always ensure that the destination drive or partition (`of=`) is of equal or greater size than the source (`if=`).

Cloning a partition

From physical disk `/dev/sda` , partition 1, to physical disk `/dev/sdb` , partition 1.

```
# dd if=/dev/sda1 of=/dev/sdb1 bs=64K conv=noerror,sync
```

Warning: If output file `of=` (`sdb1` in the example) does not exist, *dd* will create a file with this name and will start filling up your root file system!

Cloning an entire hard disk

From physical disk `/dev/sdX` to physical disk `/dev/sdY`

```
# dd if=/dev/sdX of=/dev/sdY bs=64K conv=noerror,sync
```

This will clone the entire drive, including the MBR (and therefore bootloader), all partitions, UUIDs, and data.

- `noerror` instructs *dd* to continue operation, ignoring all read errors. Default behavior for *dd* is to halt at any error.
- `sync` fills input blocks with zeroes if there were any read errors, so data offsets stay in sync.
- `bs=` sets the block size. Defaults to 512 bytes, which is the "classic" block size for hard drives since the early 1980s, but is not the most convenient. Use a bigger value, 64K or 128K. Also, please read the warning below, because there is more to this than just "block sizes" -it also influences how read errors propagate. See [1] (http://www.mail-archive.com/eug-lug@efn.org/msg12073.html) and [2] (http://blog.tdg5.com/tuning-dd-block-size/) for details and to figure out the best bs value for your use case.

Warning: The block size you specify influences how read errors are handled. Read below. For data recovery, use ddrescue.

The *dd* utility technically has an "input block size" (IBS) and an "output block size" (OBS). When you set `bs`, you effectively set both IBS and OBS. Normally, if your block size is, say, 1 MiB, *dd* will read 1024*1024 bytes and write as many bytes. But if a read error occurs, things will go wrong. Many people seem to think that *dd* will "fill up read errors with zeroes" if you use the `noerror,sync` options, but this is not what happens. *dd* will, according to documentation, fill up the OBS to IBS size *after completing its read*, which means adding zeroes at the *end* of the block. This means, for a disk, that effectively the whole 1 MiB would become messed up because of a single 512 byte read error in the beginning of the read: 12ERROR89 would become 128900000 instead of 120000089.

If you are positive that your disk does not contain any errors, you could proceed using a larger block size, which will increase the speed of your copying several fold. For example, changing bs from 512 to 64K changed copying speed from 35 MB/s to 120 MB/s on a simple Celeron 2.7 GHz system. But keep in mind that read errors on the source disk will end up as *block errors* on the destination disk, i.e. a single 512-byte read error will mess up the whole 64 KiB output block.

Tip: If you would like to view *dd* progressing, use the `status=progress` option. See dd for details.

Note:

- To regain unique UUIDs of an *ext2/3/4* filesystem, use `tune2fs /dev/sdXY -U random` on every partition.
- Partition table changes from *dd* are not registered by the kernel. To notify of changes without rebooting, use a utility like *partprobe* (part of GNU Parted).

Backing up the MBR

The MBR is stored in the the first 512 bytes of the disk. It consist of 3 parts:

1. The first 446 bytes contain the boot loader.
2. The next 64 bytes contain the partition table (4 entries of 16 bytes each, one entry for each primary partition).
3. The last 2 bytes contain an identifier

To save the MBR as `mbr.img`:

```
# dd if=/dev/sdX of=/path/to/mbr_file.img bs=512 count=1
```

To restore (be careful: this could destroy your existing partition table and with it access to all data on the disk):

```
# dd if=/path/to/mbr_file.img of=/dev/sdX
```

If you only want to restore the boot loader, but not the primary partition table entries, just restore the first 446 bytes of the MBR:

```
# dd if=/path/to/mbr_file.img of=/dev/sdX bs=446 count=1
```

To restore only the partition table, one must use:

```
# dd if=/path/to/mbr_file.img of=/dev/sdX bs=1 skip=446 count=64
```

You can also get the MBR from a full dd disk image:

```
# dd if=/path/to/disk.img of=/path/to/mbr_file.img bs=512 count=1
```

Create disk image

1. Boot from a live media.

2. Make sure no partitions are mounted from the source hard drive.

3. Mount the external HD

4. Backup the drive.

```
# dd if=/dev/sdX conv=sync,noerror bs=64K | gzip -c  > /path/to/backup.img.gz
```

If necessary (e.g. when the format of the external HD is FAT32) split the disk image in volumes (see also the *split* man pages).

```
# dd if=/dev/sdX conv=sync,noerror bs=64K | gzip -c | split -a3 -b2G - /path/to/backup.img.gz
```

If there is not enough disk space locally, you may send the image through ssh:

```
# dd if=/dev/sdX conv=sync,noerror bs=64K | gzip -c | ssh user@local dd of=backup.img.gz
```

5. Save extra information about the drive geometry necessary in order to interpret the partition table stored within the image. The most important of which is the cylinder size.

```
# fdisk -l /dev/sdX > /path/to/list_fdisk.info
```

Note: You may wish to use a block size (`bs=`) that is equal to the amount of cache on the HD you are backing up. For example, `bs=8192K` works for an 8 MiB cache. The 64 KiB mentioned in this article is better than the default `bs=512` bytes, but it will run faster with a larger `bs=` .

Restore system

To restore your system:

```
# gunzip -c /path/to/backup.img.gz | dd of=/dev/sdX
```

When the image has been split, use the following instead:

```
# cat /path/to/backup.img.gz* | gunzip -c | dd of=/dev/sdX
```

Using ddrescue

ddrescue is a tool designed for cloning and recovering data. It copies data from one file or block device (hard disc, cdrom, etc) to another, trying to rescue the good parts first in case of read errors, to maximize the recovered data.

To clone a faulty or dying drive, run ddrescue twice. First round, copy every block without read error and log the errors to rescue.log.

```
# ddrescue -f -n /dev/sdX /dev/sdY rescue.log
```

Second round, copy only the bad blocks and try 3 times to read from the source before giving up.

```
# ddrescue -d -f -r3 /dev/sdX /dev/sdY rescue.log
```

Now you can check the file system for corruption and mount the new drive.

```
# fsck -f /dev/sdY
```

Disk cloning software

These applications allow easy backup of entire filesystems and recovery in case of failure, usually in the form of a Live CD or USB drive. They contain complete system images from one or more specific points in time and are frequently used to record known good configurations.

See also Synchronization and backup programs for other applications that can take full system snapshots, among other functionality.

- **Acronis True Image** — Commercial disk cloner for Windows. It allows you to create a live (from within Windows), so you do not need a working Windows install on the actual machine to use it. After registration of the Acronis software on their website, you will be able to download a Linux-based Live CD and/or plugins for BartPE for creation of the Windows-based live CD. It can also create a WinPE Live CD based on Windows. The created ISO Live CD image by Acronis doesn't have the hybrid boot (http://www.syslinux.org/wiki/index.php/Isohybrid) ability and cannot be written to USB storage as a raw file.

http://www.acronis.com/products/trueimage/ ||

- **Arch Backup** — A trivial backup script with simple configuration.
  - Configurable compression method.
  - Multiple backup targets.

  http://code.google.com/p/archlinux-stuff/ || `arch-backup (https://aur.archlinux.org/packages/arch-backup/)`AUR

- **Clonezilla** — A disaster recovery, disk cloning, disk imaging and deployment solution.
  - Boots from live CD, USB flash drive, or PXE server.
  - Supports ext2, ext3, ext4, reiserfs, reiser4, xfs, jfs, btrfs FAT32, NTFS, HFS+ and others.
  - Uses Partclone (default), Partimage (optional), ntfsclone (optional), or dd to image or clone a partition.
  - Multicasting server to restore to many machines at once.
  - Included on the Arch Linux installation media.

  http://clonezilla.org/ || `clonezilla (https://www.archlinux.org/packages/?name=clonezilla)`

- **FSArchiver** — A safe and flexible file-system backup and deployment tool
  - Support for basic file attributes (permissions, owner, ...).
  - Support for multiple file-systems per archive.
  - Support for extended attributes (they are used by SELinux).
  - Support the basic file-system attributes (label, uuid, block-size) for all linux file-systems.
  - Support for ntfs filesystems (http://www.fsarchiver.org/Cloning-ntfs) (ability to create flexible clones of a Windows partitions).
  - Checksumming of everything which is written in the archive (headers, data blocks, whole files).
  - Ability to restore an archive which is corrupt (it will just skip the current file).
  - Multi-threaded lzo, gzip, bzip2, lzma compression.
  - Support for splitting large archives into several files with a fixed maximum size.
  - Encryption of the archive using a password. Based on blowfish from libcrypto from OpenSSL.
  - Support backup of a mounted root filesystem (-A option).
  - Can be found on the System Rescue CD (http://www.sysresccd.org/Main_Page).

  http://www.fsarchiver.org/Main_Page || `fsarchiver (https://www.archlinux.org/packages/?name=fsarchiver)`

- **Mondo Rescue** — A disaster recovery solution to create backup media that can be used to redeploy the damaged system.
  - Image-based backups, supporting Linux/Windows.
  - Compression rate is adjustable.
  - Can backup live systems (without having to halt it).
  - Can split image over many files.
  - Supports booting to a Live CD to perform a full restore.
  - Can backup/restore over NFS, from CDs, tape drives and and other media.
  - Can verify backups.

  http://www.mondorescue.org/ || `mondo (https://aur.archlinux.org/packages/mondo/)`AUR

- **Partclone** — A tool that can be used to back up and restore a partition while considering only used blocks.
  - Supports *ext2*, *ext3*, *ext4*, *hfs+*, *reiserfs*, *reiser4*, *btrfs*, *vmfs3*, *vmfs5*, *xfs*, *jfs*, *ufs*, *ntfs*, *fat(12/16/32)*, *exfat*.
  - Supports compression.
  - Optionally, an *ncurses* interface can be used.

  http://partclone.org/ || `partclone (https://www.archlinux.org/packages/?name=partclone)`

- **PartedMagic** — Live cd/usb with PartImage and other recovery tools.

  http://partedmagic.com/doku.php?id=start ||

- **Partimage** — An *ncurses* disk cloning utility for Linux/UNIX environments.
  - Has a Live CD.
  - Supports the most popular filesystems on Linux, Windows and Mac OS.
  - Compression.
  - Saving to multiple CDs or DVDs or across a network using Samba/NFS.

  http://www.partimage.org/Main_Page || `partimage (https://www.archlinux.org/packages/?name=partimage)`

- **Q7Z** — P7Zip GUI for Linux, which attempts to simplify data compression and backup. It can create the following archive types: 7z, BZip2, Zip, GZip, Tar.
  - Updates existing archives quickly.
  - Backup multiple folders to a storage location.
  - Create or extract protected archives.
  - Lessen effort by using archiving profiles and lists.

  http://k7z.sourceforge.net/ || `q7z (https://aur.archlinux.org/packages/q7z/)`AUR

- **Redo Backup and Recovery** — A backup and disaster recovery application that runs from a bootable Linux CD image.
  - Is capable of bare-metal backup and recovery of disk partitions.
  - Uses xPUD (http://www.xpud.org/) and Partclone for the backend.

  http://www.redobackup.org/ ||

- **System Tar & Restore** — Backup and Restore your system using tar or Transfer it with rsync
  - CLI and Dialog interfaces
  - Easy backup and restore wizards
  - Creates *.tar.gz*, *.tar.bz2*, *.tar.xz* or *.tar* archives
  - Supports openssl / gpg encryption
  - Uses rsync to transfer a running system
  - Supports Grub2, Syslinux, EFISTUB/efibootmgr and Systemd/bootctl

  https://github.com/tritonas00/system-tar-and-restore || `system-tar-and-restore (https://aur.archlinux.org/packages/system-tar-and-restore/)`AUR

dd spin-offs

Other *dd*-like programs feature periodical status output, e.g. a simple progress bar.

dcfldd

dcfldd (https://www.archlinux.org/packages/?name=dcfldd) is an enhanced version of dd with features useful for forensics and security. It accepts most of dd's parameters and includes status output. The last stable version of dcfldd was released on December 19, 2006.[3] (http://dcfldd.sourceforge.net/)

ddrescue

GNU ddrescue (https://www.archlinux.org/packages/?name=ddrescue) is a data recovery tool. It is capable of ignoring read errors, which is a useless feature for disk wiping in almost any case. See the official manual (http://www.gnu.org/software/ddrescue/manual/ddrescue_manual.html) for details.

See also

- Wikipedia:List of disk cloning software
- Arch Linux forum thread (https://bbs.archlinux.org/viewtopic.php?id=4329)

Retrieved from "https://wiki.archlinux.org/index.php?title=Disk_cloning&oldid=434977"

Categories: Data compression and archiving | System recovery

- This page was last modified on 14 May 2016, at 11:45.
-