Configure Software RAID on Linux

It's a common scenario to use software RAID on Linux virtual machines in Azure to present multiple attached data disks as a single RAID device. Typically this can be used to improve performance and allow for improved throughput compared to using just a single disk.

## **Attaching data disks**

Two or more empty data disks are needed to configure a RAID device. The primary reason for creating a RAID device is to improve performance of your disk IO. Based on your IO needs, you can choose to attach disks that are stored in our Standard Storage, with up to 500 IO/ps per disk or our Premium storage with up to 5000 IO/ps per disk. This article does not go into detail on how to provision and attach data disks to a Linux virtual machine. See the Microsoft Azure article [attach a disk](https://docs.microsoft.com/en-us/azure/virtual-machines/linux/add-disk?toc=%2fazure%2fvirtual-machines%2flinux%2ftoc.json) for detailed instructions on how to attach an empty data disk to a Linux virtual machine on Azure.

## **Install the mdadm utility**

* **Ubuntu**

bashCopy

sudo apt-get update

sudo apt-get install mdadm

* **CentOS & Oracle Linux**

bashCopy

sudo yum install mdadm

* **SLES and openSUSE**

bashCopy

zypper install mdadm

## **Create the disk partitions**

In this example, we create a single disk partition on /dev/sdc. The new disk partition will be called /dev/sdc1.

1. Start fdisk to begin creating partitions

bashCopy

sudo fdisk /dev/sdc

Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel

Building a new DOS disklabel with disk identifier 0xa34cb70c.

Changes will remain in memory only, until you decide to write them.

After that, of course, the previous content won't be recoverable.

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to

switch off the mode (command 'c') and change display units to

sectors (command 'u').

1. Press 'n' at the prompt to create a **n**ew partition:

bashCopy

Command (m for help): n

1. Next, press 'p' to create a **p**rimary partition:

bashCopy

Command action

e extended

p primary partition (1-4)

1. Press '1' to select partition number 1:

bashCopy

Partition number (1-4): 1

1. Select the starting point of the new partition, or press <enter> to accept the default to place the partition at the beginning of the free space on the drive:

bashCopy

First cylinder (1-1305, default 1):

Using default value 1

1. Select the size of the partition, for example type '+10G' to create a 10 gigabyte partition. Or, press <enter> create a single partition that spans the entire drive:

bashCopy

Last cylinder, +cylinders or +size{K,M,G} (1-1305, default 1305):

Using default value 1305

1. Next, change the ID and **t**ype of the partition from the default ID '83' (Linux) to ID 'fd' (Linux raid auto):

bashCopy

Command (m for help): t

Selected partition 1

Hex code (type L to list codes): fd

1. Finally, write the partition table to the drive and exit fdisk:

bashCopy

Command (m for help): w

The partition table has been altered!

## **Create the RAID array**

1. The following example will "stripe" (RAID level 0) three partitions located on three separate data disks (sdc1, sdd1, sde1). After running this command a new RAID device called **/dev/md127** is created. Also note that if these data disks we previously part of another defunct RAID array it may be necessary to add the --force parameter to the mdadm command:

bashCopy

sudo mdadm --create /dev/md127 --level 0 --raid-devices 3 \

/dev/sdc1 /dev/sdd1 /dev/sde1

1. Create the file system on the new RAID device

a. **CentOS, Oracle Linux, SLES 12, openSUSE, and Ubuntu**

bashCopy

sudo mkfs -t ext4 /dev/md127

b. **SLES 11**

bashCopy

sudo mkfs -t ext3 /dev/md127

c. **SLES 11** - enable boot.md and create mdadm.conf

bashCopy

sudo -i chkconfig --add boot.md

sudo echo 'DEVICE /dev/sd\*[0-9]' >> /etc/mdadm.conf

Note

A reboot may be required after making these changes on SUSE systems. This step is not required on SLES 12.

## **Add the new file system to /etc/fstab**

Important

Improperly editing the /etc/fstab file could result in an unbootable system. If unsure, refer to the distribution's documentation for information on how to properly edit this file. It is also recommended that a backup of the /etc/fstab file is created before editing.

1. Create the desired mount point for your new file system, for example:

bashCopy

sudo mkdir /data

1. When editing /etc/fstab, the **UUID** should be used to reference the file system rather than the device name. Use the blkid utility to determine the UUID for the new file system:

bashCopy

sudo /sbin/blkid

...........

/dev/md127: UUID="aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee" TYPE="ext4"

1. Open /etc/fstab in a text editor and add an entry for the new file system, for example:

bashCopy

UUID=aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee /data ext4 defaults 0 2

Or on **SLES 11**:

bashCopy

/dev/disk/by-uuid/aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee /data ext3 defaults 0 2

Then, save and close /etc/fstab.

1. Test that the /etc/fstab entry is correct:

bashCopy

sudo mount -a

If this command results in an error message, please check the syntax in the /etc/fstab file.

Next run the mount command to ensure the file system is mounted:

bashCopy

mount

.................

/dev/md127 on /data type ext4 (rw)

1. (Optional) Failsafe Boot Parameters

**fstab configuration**

Many distributions include either the nobootwait or nofail mount parameters that may be added to the /etc/fstab file. These parameters allow for failures when mounting a particular file system and allow the Linux system to continue to boot even if it is unable to properly mount the RAID file system. Refer to your distribution's documentation for more information on these parameters.

Example (Ubuntu):

bashCopy

UUID=aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee /data ext4 defaults,nobootwait 0 2

**Linux boot parameters**

In addition to the above parameters, the kernel parameter "bootdegraded=true" can allow the system to boot even if the RAID is perceived as damaged or degraded, for example if a data drive is inadvertently removed from the virtual machine. By default this could also result in a non-bootable system.

Please refer to your distribution's documentation on how to properly edit kernel parameters. For example, in many distributions (CentOS, Oracle Linux, SLES 11) these parameters may be added manually to the "/boot/grub/menu.lst" file. On Ubuntu this parameter can be added to the GRUB\_CMDLINE\_LINUX\_DEFAULT variable on "/etc/default/grub".

## **TRIM/UNMAP support**

Some Linux kernels support TRIM/UNMAP operations to discard unused blocks on the disk. These operations are primarily useful in standard storage to inform Azure that deleted pages are no longer valid and can be discarded. Discarding pages can save cost if you create large files and then delete them.

Note

RAID may not issue discard commands if the chunk size for the array is set to less than the default (512KB). This is because the unmap granularity on the Host is also 512KB. If you modified the array's chunk size via mdadm's --chunk= parameter, then TRIM/unmap requests may be ignored by the kernel.

There are two ways to enable TRIM support in your Linux VM. As usual, consult your distribution for the recommended approach:

* Use the discard mount option in /etc/fstab, for example:

bashCopy

UUID=aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee /data ext4 defaults,discard 0 2

* In some cases the discard option may have performance implications. Alternatively, you can run the fstrim command manually from the command line, or add it to your crontab to run regularly:

**Ubuntu**

bashCopy

# sudo apt-get install util-linux

# sudo fstrim /data

**RHEL/CentOS**

bashCopy

# sudo yum install util-linux

# sudo fstrim /data