How to install and use docker with btrfs on CentOS 7

In this tutorial I want to show how you can install docker on CentOS 7 and using a BTRFS partition as underlying store.

Install docker

```
  $ yum update  Geladene Plugins: fastestmirror, priorities  Loading m
irror speeds from cached hostfile   * base: centos.mirror.sharkservers
.co.uk   * extras: centosmirror.netcup.net   * updates: mirror.softacu
lous.com  10055 packages excluded due to repository priority protectio
ns  No packages marked for update
```

If you see a line like *10055 packages excluded due to repository priority protections*, then a yum configuration is needed. The message means some packages are held by more than one repository. The priorities plugin choose packages from the highest-priority repository, excluding duplicate entries from other repos. If you don't update first, then *yum install docker* will not work because of dependency problems. To make this to work you have to edit /etc/yum/pluginconf.d/priorities.conf and change the content to:

```
  [main]  enabled=0  check_obsoletes=1
```

Now try again to install docker:

```
  $ yum update  $ yum install docker
```

Now you can enable docker to start on boot time:

```
  $ systemctl enable docker.service  ln -s '/usr/lib/systemd/system/do
cker.service' '/etc/systemd/system/multi-
```

```
user.target.wants/docker.service'
```

Let's check the status of the service:

```
  $ systemctl status docker.service  docker.service - Docker Applicati
on Container Engine     Loaded: loaded (/usr/lib/systemd/system/docker
.service; enabled)     Active: inactive (dead)       Docs: http://docs
.docker.com
```

Docker is still not running. Now reboot your machine or start docker with:

```
  $ systemctl start docker.service
```

Afterwards we want to see some docker information to confirm everything works as expected:

```
  $ docker info  Containers
: 0  Images: 0
Storage Driver: devicemapper
   Pool Name: docker-253:1-683-pool   Pool Blocksize: 65.54 kB   Data
file: /var/lib/docker/devicemapper/devicemapper/data   Metadata file:
/var/lib/docker/devicemapper/devicemapper/metadata   Data Space Used:
307.2 MB   Data Space Total: 107.4 GB   Metadata Space Used: 733.2 kB
  Metadata Space Total: 2.147 GB   Library Version: 1.02.84-RHEL7 (201
4-03-26)  Execution Driver: native-0.2  Kernel Version: 3.10.0-123.el7
.x86_64  Operating System: CentOS Linux 7 (Core)
```

Device mapper thin provisioning

By default docker uses the device mapper thin provisioning to manage Docker containers if AUFS is not available on the operating system. This is the case if you install docker on a CentOS 7 system for example. For the default storage type "device mapper" no additional configuration is needed. The drawback all the containers are stored in the root partition under /var/lib/docker. Enter the following to see more information on a default docker system.

```
  $ sudo lsblk  NAME                        MAJ:MIN RM  SIZE RO TYPE
MOUNTPOINT  fd0                           2:0    1    4K  0 disk   sd
a                           8:0    0    8G  0 disk   ??sda1
               8:1    0  500M  0 part /boot  ??sda2
```

```
      8:2    0  7.5G  0 part     ??centos-swap              253:0    0
  820M  0 lvm  [SWAP]    ??centos-root              253:1    0  6.7G
0 lvm  /  sr0                         11:0    1 1024M  0 rom     loop
0                          7:0    0  100G  0 loop   ??docker-253:1-258
57769-pool 253:2    0  100G  0 dm      loop1                           7
:1    0    2G  0 loop   ??docker-253:1-25857769-pool 253:2    0  100G
 0 dm
```

Notice the loopback mounted device. You can use docker in this way on a developer machine but don't use it on a production system. Furthermore the size is 100GB maximum. The real disk usage is less, only so much your docker containers currently need. If you need a store more than 100GB or a faster one you can use a real device instead of a file-backed loop device. More about the device mapper can be found on Jérôme Petazzoni's blog: Resizing Docker containers with the Device Mapper plugin.

Docker and btrfs

Btrfs is a new copy on write (CoW) filesystem for Linux, you can find more inforamtion about btrfs here. In the redhat developer blog you can read: btrfs seems the most natural fit for Docker. If you install a new centos operating system make sure to create a partition with the Btrfs filesystem and mount it to /var/lib/docker. If you have installed your system already then use the following commands (make sure vda3 or another empty partition exists):

```
  $ systemctl stop docker  $ rm -rf /var/lib/docker  $ yum install -y
btrfs-progs btrfs-progs-devel  $ mkfs.btrfs -f /dev/vda3 (caution this
 will delete all the datas on /dev/vda3!)  $ mkdir /var/lib/docker  $
echo "/dev/vda3 /var/lib/docker btrfs defaults 0 0" >> /etc/fstab  $ m
ount -a
```

If you are not sure about your harddisk partitions you can use the following commands to show, delete or create new partitions:

```
  $ cat /proc/partitions  major minor  #blocks  name     253        0
1023410176 vda   253         1    7168000 vda1   253         2    102400
0 vda2   253         3  419430400 vda3   253         4  595786752 vda4
  11        0    1048575 sr0
```

```
  # df -h  Dateisystem    Größe Benutzt Verf. Verw% Eingehängt auf  /d
ev/vda1       6,7G    1,2G  5,1G   19% /  devtmpfs        7,8G       0
```

```
  7,8G    0% /dev  tmpfs           7,8G       0  7,8G    0% /dev/shm
tmpfs           7,8G    8,3M  7,8G    1% /run  tmpfs           7,8G
    0  7,8G    0% /sys/fs/cgroup  /dev/vda3       400G    512K  398G
  1% /var/lib/docker  /dev/vda4       560G     73M  531G    1% /data
```

```
  fdisk /dev/vda
```

Now adapt the docker configuration for usingn btrfs, it should look similar to the following afterwards:

```
  # /etc/sysconfig/docker    # Modify these options if you want to cha
nge the way the docker daemon runs  #OPTIONS=--selinux-enabled -H fd:/
/  OPTIONS=-H fd:// -D -s btrfs    # Location used for temporary files
, such as those created by  # docker load and build operations. Defaul
t is /var/lib/docker/tmp  # Can be overriden by setting the following
environment variable.  # DOCKER_TMPDIR=/var/tmp
```

Because btrfs does not currently support SELinux the OPTIONS line don't have the switch --selinux-enabled anymore. -s btrfs forces the Docker runtime to use the btrfs storage driver. That's all now start the docker daemon and check the status by typing:

```
  $ systemctl start docker  $ systemctl status docker
```

Now confirm the new store type with:

```
  $ docker info  Containers
: 0  Images: 0  Storage Driver: btrfs
  Execution Driver: native-0.2  Kernel Version: 3.10.0-123.13.2.el7.x8
6_64  Operating System: CentOS Linux 7 (Core)  Debug mode (server): tr
ue  Debug mode (client): false  Fds: 10  Goroutines: 11  EventsListene
rs: 0  Init SHA1: c906504aa058139c1d0569ecd0aa5f462a73440f  Init Path:
 /usr/libexec/docker/dockerinit
```

Now try to pull a docker image and run a container from it.

```
  $ docker pull busybox  $ docker images  $ docker run -it --rm busybo
```

x

Type exit and enter, if you want stop the busybox container. More docker commands can you find here.

Now I hope you are able to use docker with btrfs. Thanks for reading my blog and drop me a mail, if you have any questions.