Docker: Backup

In this Chapter we will Backup the Docker- Volumes and the Data of the home- Directory of Docker, that is important. We will NOT do a full Backup of the Server, while Docker can only archive what is accessable by the user docker (rootless!).

Further we **will need a working Nextcloud- AIO with working Nextcloud- Backup preconfigured** - so if you have not set up NC-AIO, start there.

The benefit of this is, that all the contents of docker will be backed up with the right acces-rights and configurations an we can rely on the functionality of NC-AIO here.

Setup Scripts

Create the Directory for your Backup- Scripts, e.g. /home/docker/docker_compose/backup

Inside that directory, create a file named „`additional_backup_directories_initial`“ and insert the content:

```
/home/docker/docker_compose
/home/docker/.config
/home/docker/.docker
/home/docker/bin
```

These are the basic Directories of the User.

Than, create a script named „docker_backup_all.sh“, make it executable an insert the Content:

```
#!/bin/bash
# Will make a Backup of Docker and all Volumes by calling NC-AIO Backup
DIRECTORYFILE='additional_backup_directories'
#Services and order to stop
declare -a SERVICE=("caddy" "ipa" "mariadb" "portainer")

# Set working dir
cd /home/docker/docker_compose/backup
# Get all Directories to backup
cat  ${DIRECTORYFILE}_initial> ${DIRECTORYFILE}
```

```
find "/home/docker/.local/share/docker/volumes" -maxdepth 1 -type d | grep pcserver2023>> ${DIRECTORYFILE}

# Remove Database- Directories
sed -i '/pgsql_data/d' ${DIRECTORYFILE}

# Make Postgres-Backup of Authentik
../authentik/docker_backup_authentik_db.sh

# Copy the file to NC-AIO
echo "The following Directories will be backup up ADDITIONALLY to Nextcloud AIO:"
cat ${DIRECTORYFILE}
docker cp ${DIRECTORYFILE} nextcloud-aio-mastercontainer:/mnt/docker-aio-config/data
docker exec nextcloud-aio-mastercontainer chown www-data:www-data /mnt/docker-aio-config/data/additional_backup_directories
docker exec nextcloud-aio-mastercontainer chmod o+r /mnt/docker-aio-config/data/additional_backup_directories

# Now stop all containers NOT beeing part of NC
for i in "${SERVICE[@]}"; do
   echo "Stopping $i"
   cd /home/docker/docker_compose/$i
   docker compose down
   sleep 1
done

echo "Now backup is done by AIO"
docker exec -it --env DAILY_BACKUP=1 nextcloud-aio-mastercontainer /daily-backup.sh
echo "The Log is:"
docker logs nextcloud-aio-borgbackup

# At this Point all Services should be shutdown despite the AIO-Mastercontainer
# while some services tend to hang sometimes, i now want to completely shutdown everything
/home/docker/bin/docker stop -t 180 $(/home/docker/bin/docker ps -a -q)
/home/docker/bin/docker rm $(/home/docker/bin/docker ps -a -q)
# and even kill docker-service and restart again
systemctl --user restart docker
# now, restart first NC-AIO Mastercontainer for further jobs
cd /home/docker/docker_compose/nextcloud_aio
/home/docker/bin/docker compose up -d
```

```
# Start the NC-AOI Services by AIO
sleep 2
docker exec -it --env START_CONTAINERS=1 nextcloud-aio-mastercontainer /daily-backup.sh

# Start Services in reverse order
for ((i=${#SERVICE[@]}-1; i>=0; i--)); do
   echo "Starting ${SERVICE[i]}"
   cd /home/docker/docker_compose/${SERVICE[i]}
   docker compose up -d
   sleep 1
done
```

Thats the whole Backup-Script.

Cronjob

Frist, create another file in that Directory e.g. /home/docker/docker_compose/backup/docker_backup_cron.sh with the contents:

```
#!/bin/bash
cd /home/docker/docker_compose/backup
./docker_backup_all.sh> docker_backup_cron.log 2>&1
cat docker_backup_cron.log
```

Use `crontab -e` to create the cronjob for user docker on your host:

```
0 3 * * * /home/docker/docker_compose/backup/docker_backup_cron.sh
```

Thats all. Make sure, you recieve Mails for this user.

Restore and UserIDs in the Borg-Backups

Mind, that Borg will store the UID/GID of some file/directory from inside running docker- container. That means, that if the file on the host has some SubUID, like e.g. UID = 165568, the file seen from inside the container can be UID = 33, which could be the container- user named „www-data“.

So you cannot restore the Files on your Host and hope, that this will work. To restore these archives, you need (again) a running Nextcloud- AIO and restore the files from that borg- container, that is included there. **AGAIN, THIS IS VERY IMPORTANT: DO NOT TRY TO OPEN BORG-BACKUPS ON THE HOST AND RESTORE THEM FROM THERE To YOUR DOCKER- VOLUMES !!! YOU WILL MESS UP THE PERMISSIONS IN THE CONTAINER !**

And DO NOT THINK YOU CAN SET USERS MANUALLY LATER on the Hosts Volumes ! → some containers, like FreeIPA do include non-relative symlinks in their data, even pointing to Files on your Host when looking at them from your Host. If you chown them, than HOST SYSTEMFILES will get messed up, maybe rendering your whole Server unusable!

For making restore of Docker- Volumes easy, check out this community-container:

https://github.com/nextcloud/all-in-one/tree/main/community-containers/borgbackup-viewer

This will do the mounts and will make it easy to copy files from borg backup to the volumes.