HOMEWORK 2

Answer all questions from practical assignment 2.

Section(s):

Play With Snapshots And Images

Question: Open `http://195.201.14.XXX` with a browser again. What do you see and what do you think happened?
Answer: First we created an `Apache2` web server in our container. That allowed us to visit its `localhost` and see the *Welcome* page of the properly set-up `Apache`. Then we created a snapshot of that container, where in our case it was called `first-snapshot`. We went about our business and altered some files, more specifically, we altered the landing page of the `Apache2` web server. At this point we decided to stop our container, in order for us to restore it from the `snapshot` we just created. Now when we look at the *Welcome* page again, we can see that our changes were *lost*. Basically, we tested how `copy-on-write` works. By doing the `restore`, the pointer (i.e. `first-snapshot`) to our container just changed the location in pointed to. That is why we lost our changes, because when we *saved* the modified `html`, it was saved to a place where the snapshot did not point to.

Question: What do you think can go wrong when restoring a container from a snapshot?
Answer: Several things can go wrong in such a situation. First of all, if we do not create *regular* snapshots of our containers, we are setting up ourselves for a nasty surprise of *"If you revert to a past snapshot, all you current changes will disappear"*. But we also might put ourselves in a false sense of security, thinking that this is a valuable form of backup, but in reality it's not. This means that even if we are keeping track of what we `snapshot`, we still might revert to a previous state that makes us loose valuable data. This kind-of relates to the *git-panic* situations.

Question: Can you describe what we did exactly with the above set of commands?
Answer: First of, we are creating a new container called `dev1`, by specifying the operating system we want the container to have. Since we've already done that, this time the operation is quicker. Then we issued a `destroy` command on the newly created container, thus removing any *datasets* in it. But because we *saved* our data set from the previous container, under the name `dev1.raw`, we can use the `receive` command in order to restore our file system to its previous state. Finally we have to tell `zfs` where to mount for the specific container. This is done through the `set mountpoint` command.

Question: What do you think we can use this approach for?
Answer: One of the uses of this approach is that we can restore a container from backup incredibly fast. We did not wait for anything, and we got our previous state as it was (given that we did have a backup ready). This also means that we can create several identical containers and jump start a task fairly quickly, without having to configure/re-configure anything.

Extra References

How do snapshot backups work
Copy-on-write
What is storage snapshot technology
Crash course of ZFS

Martin Nestorov