Mar 8 2016

Share:
(https://twitter.com/intent/tweet?text=Cobbler in a Docker Container&url=http://container-solutions.com/cobbler-in-a-docker-container/)
(https://www.facebook.com/sharer/sharer.php?u=http://container-solutions.com/cobbler-in-a-docker-container/)
(mailto:?subject=Check this out&body=This: http://container-solutions.com/cobbler-in-a-docker-container/)

Cobbler in a Docker Container

by Thijs Schnitger (http://container-solutions.com/author/thijs/) 3 (http://container-solutions.com/cobbler-in-a-docker-container#comments) Submit

For one of our projects we found ourselves in need of a PXE server. In order to make proper use of a PXE server you will quickly find you need to change settings in your DHCP server. So we figured we'd probably needed one of those as well. We tried to roll our own but soon enough we came across a package that does all this combined. It's called Cobbler (https://cobbler.github.io/).

Also being somewhat biased, we would prefer to run Cobbler in a container. That comes with some challenges in itself. If you search for Cobbler on the Docker Hub, there are 10 hits. 4 of them are not automated builds, which means you can't really tell what's in them. None of the others seem to be using systemd, which makes it kind of hard for Cobbler to manage the services, which in turn makes it rather impossible to change the config of a running Cobbler container.

So the obvious thing to do is to roll our own. Which we did.

Building the Docker image

Docker is primarily meant to run a single process, but Cobbler consists of mulitple services that you cannot easily separate. There's cobblerd itself, httpd for the webfrontend, tftpd for serving the PXE images, dhcpd serving ip addresses, and possibly dns for resolving hostnames. Cobbler wants control over all of these services, i.e. after you make changes to the configuration it needs to restart the services so they can pick up the changes. This is why it makes sense to put all these things in a single container, although it makes for a bit of a bloated one. Restarting services requires an init system such as systemd, and luckily the official Centos image is able to run with systemd. Details on how to use it can be found on the CentOS Docker Hub page (https://hub.docker.com/_/centos/). So the beginning of the Dockerfile will look something like this:

```
FROM centos:7.2.1511

MAINTAINER thijs.schnitger@container-solutions.com
RUN (cd /lib/systemd/system/sysinit.target.wants/; for i in *; do [ $i == systemd-tmpfil
rm -f /lib/systemd/system/multi-user.target.wants/*;\
rm -f /etc/systemd/system/*.wants/*;\
rm -f /lib/systemd/system/local-fs.target.wants/*; \
rm -f /lib/systemd/system/sockets.target.wants/*udev*; \
rm -f /lib/systemd/system/sockets.target.wants/*initctl*; \
rm -f /lib/systemd/system/basic.target.wants/*;\
rm -f /lib/systemd/system/anaconda.target.wants/*;
VOLUME [ “/sys/fs/cgroup” ]
```

Next we install the EPEL (https://fedoraproject.org/wiki/EPEL) repositories that contain the Cobbler packages we need. Once we add EPEL we can install Cobbler and the rest of the necessary packages.

```
RUN yum -y install epel-release
RUN yum -y install cobbler cobbler-web dhcp bind syslinux pykickstart
```

We need to enable the services in systemd.

```
RUN systemctl enable cobblerd
RUN systemctl enable httpd
RUN systemctl enable dhcpd
```

Then we change the tftp xinetd config file to enable the tftp service.

```
# enable tftp
RUN sed -i -e 's/\(^.*disable.*=\) yes/\1 no/' /etc/xinetd.d/tftp
```

And we make sure the rsync file exists, or else Cobbler will fail to start.

```
# create rsync file
RUN touch /etc/xinetd.d/rsync
```

We expose the ports we plan to use:

```
EXPOSE 69
EXPOSE 80
EXPOSE 443
EXPOSE 25151
```

And we end by invoking init

```
CMD ["/sbin/init"]
```

Once we have declared the Dockerfile, we build it with `docker build -t cobbler .` and we should be good to go. But not quite.

Running the Container

We can't just run it, we need to specify `--privileged` because we're using systemd which needs elevated capabilities, like CAP_SET_FILE.

We also want to use the network stack of the host, so Cobbler will listen on and offer ip addresses on a subnet that the host is connected to. If it would use it's own network stack Cobbler would only be able to issue addresses in the private docker subnet, which would not really make any sense because every container already gets an address from the docker engine.

Another thing we want is to save the Cobbler configuration and the imported OS distributions in between restarts of the container. For this we need to mount a bunch of volumes. Lastly we also need to mount an iso inside the container to be able to import it's contents into Cobbler. So the resulting `docker run` command is gonna look like this:

```
docker run \
-d \
--privileged \
--net host \
-v /sys/fs/cgroup:/sys/fs/cgroup:ro \
-v etc/cobbler/settings:/etc/cobbler/settings \
-v etc/cobbler/dhcp.template:/etc/cobbler/dhcp.template \
-v var/www/cobbler/images:/var/www/cobbler/images \
-v var/www/cobbler/ks_mirror:/var/www/cobbler/ks_mirror \
-v var/www/cobbler/links:/var/www/cobbler/links \
-v var/lib/cobbler/config:/var/lib/cobbler/config \
-v var/lib/tftpboot:/var/lib/tftpboot \
-v dist/centos:/mnt:ro \
-p 69:69 \
-p 80:80 \
-p 443:443 \
-p 25151:25151 \
--name cobbler cobbler
```

Before we can run though, we need to configure some files. Get these files by starting the Cobbler container without the volumes mounted, and use `docker cp` to copy over at least the `settings` and `dhcp_template` files from `/etc/cobbler` inside the container to the host. Put them in a directory `etc/cobbler` under the current dir, which would be where the Dockerfile is.

Next, edit the `settings` file, change the value for `manage_dhcp` to `1` and enter the ip address where your host will be listening on in both the `next_server` and `server` variables.

In the `dhcp_template` file, configure the subnet details. Be sure to use a subnet that your host can actually connect to, or Cobbler will fail to start later. Debugging this is kind of hard and cost me a lot of time. There are no logs written anywhere and you can only find out what's wrong by checking `systemctl status <servicename>` or `journalctl -xe`. Once you have the addresses sorted though you should be able to use the run command listed above.

Importing a distribution

You see we used a CentOS iso for importing into Cobbler, we mounted it on `dist/centos` on our host and attached that as a volume to `/mnt` inside the container. I found the `cobbler import` command fails if you use it with any other path than `/mnt`. The complete mount command is:

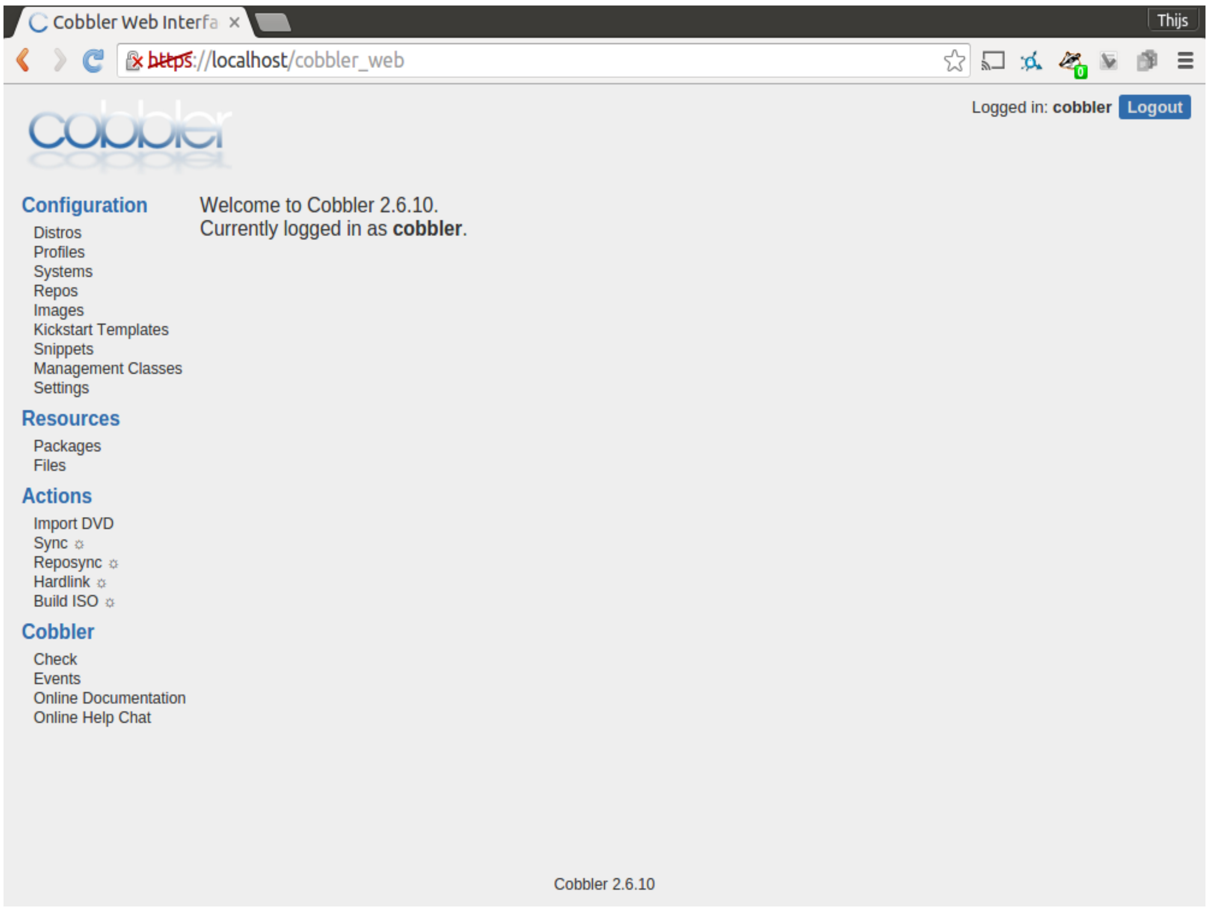
```
sudo mount -t iso9660 -o loop,ro -v dist/centos.iso dist/centos
```

and for importing the distribution:

```
docker exec -it cobbler cobbler import --name=centos7 --arch=x86_64 --path=/mnt
```

Checking the web interface

After you have started the container, you should be able to go to the web interface under https://localhost/cobbler_web (https://localhost/cobbler_web).



Welcome to Cobbler 2.6.10.
Currently logged in as **cobbler**.

Cobbler 2.6.10

Login using `cobbler` as the username and `cobbler` as the password. If the web interface doesn't respond, exec into the container and check the `systemctl status` for `dhcpd`, `cobblerd` and `httpd` and also `journalctl -xe`.

Seeing Cobbler in action

For my test setup i used a Virtualbox VM to connect to the Cobbler server. I created a host-only network and attached an empty client to that. I configured the ip address of the `vboxnet` interface in the `settings` and `dhcp_template` files.

Last, not unimportant step is to create a system in Cobbler that actually uses the distribution you imported. Configuration is straightforward, but be sure to match the MAC-address assigned to the VM by VirtualBox.
Once the vm boots, press F12 and select the LAN boot device. You should see the VM booting into the CentOS installer.

Final thoughts

All the code from this post can be found on our Github project page (https://github.com/ContainerSolutions/docker-cobbler). I also tried to get Docker Hub to do an automated build (https://hub.docker.com/r/containersol/docker-cobbler/) but due to an issue with Docker Hub (see here (https://github.com/docker/docker/issues/6980) and here (https://github.com/CentOS/CentOS-Dockerfiles/issues/46)), installing httpd in a CentOS machine doesn't work on the Docker Hub. It appears to be related to using aufs as the storage backend. For now, just build it on your local machine.

Bio **Latest Posts**

Thijs Schnitger
Senior Engineer at Container Solutions (http://container-solutions.com)

Thijs is a Systems Engineer specialized in all things Linux. At Container Solutions he mainly focuses on programmable infrastructure, when he's not busy lending a hand. His years of experience with development, delivery and operations make him an allround application debugger and problem solver.

(http://twitter.com/thijsschnitger)

(https://nl.linkedin.com/in/thijsschnitger)

3 Comments

Alastair Munro
July 29, 2016 at 2:45 am (/cobbler-in-a-docker-container/#comment-393)

Awesome. Just what I was looking for. Will give it a whirl when I get a chance. Thanks for sharing.

Reply (http://container-solutions.com/cobbler-in-a-docker-container/?replytocom=393#respond)

Mike Schmidt
August 14, 2016 at 5:57 am (/cobbler-in-a-docker-container/#comment-399)

This works quite well. Thank you.

A few notes:
– Please add a reference to this blog in the README. Anyone looking at the github project might have trouble figuring things out.
– Also note that it is important to go through the settings and adjust them, especially the ip address.
– cobbler sync won't work because sync_post_restart_services calls "service dhcpd restart" and /sbin/service is not there. This would work if not in a container, because /sbin/service is there and automatically gets forwarded to systemctl. To me, this is a bug in cobbler, but it can be remedied by installing initscripts, although I don't know what else will break as a consequence. Maybe it is easier to let cobbler sync crash on the calls to restart dhcpd and named,)or simply disable that in settings) and simply restart the container at that point.

Reply (http://container-solutions.com/cobbler-in-a-docker-container/?replytocom=399#respond)

Thijs Schnitger
August 16, 2016 at 4:00 pm (/cobbler-in-a-docker-container/#comment-402)

Glad you like it! I updated the README a bit to explain things.
On the sync issue, I think I just let it crash and restarted the dhcp and named services manually when I needed. If I need to use it again, I'll definitely look at adding initscripts.
Thanks!

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Post Comment

Related Posts

Running Docker Containers with Systemd

(http://container-solutions.com/running-docker-containers-with-systemd/)

READ MORE (HTTP://CONTAINER-SOLUTIONS.COM/RUNNING-DOCKER-CONTAINERS-WITH-SYSTEMD/)

tp://container-solutions.com/running-docker-containers-with-systemd/)

Puppet in Docker

(http://container-solutions.com/puppet-in-docker/)

READ MORE (HTTP://CONTAINER-SOLUTIONS.COM/PUPPET-IN-DOCKER/)

tp://container-solutions.com/puppet-in-docker/)

Terraform provider for Cobbler (http://container-solutions.com/terraform-provider-for-cobbler/)

READ MORE (HTTP://CONTAINER-SOLUTIONS.COM/TERRAFORM-PROVIDER-FOR-COBBLER/)

:tp://container-solutions.com/terraform-provider-for-cobbler/)

:tp://container-solutions.com/set-the-ip-of-the-docker-bridge-with-systemd/)

Set the ip of the Docker bridge with Systemd (http://container-solutions.com/set-the-ip-of-the-docker-bridge-with-systemd/)

READ MORE (HTTP://CONTAINER-SOLUTIONS.COM/SET-THE-IP-OF-THE-DOCKER-BRIDGE-WITH-SYSTEMD/)

:tp://container-solutions.com/user-access-manager-application-on-mantl/)

User Access Manager application on Mantl (http://container-solutions.com/user-access-manager-application-on-mantl/)

READ MORE (HTTP://CONTAINER-SOLUTIONS.COM/USER-ACCESS-MANAGER-APPLICATION-ON-MANTL/)

tp://container-solutions.com/hosted-docker-enabled-ci-tooling-show-down/)

Hosted Docker-Enabled CI Tooling – Show down! (http://container-solutions.com/hosted-docker-enabled-ci-tooling-show-down/)

READ MORE (HTTP://CONTAINER-SOLUTIONS.COM/HOSTED-DOCKER-ENABLED-CI-TOOLING-SHOW-DOWN/)

Container Solutions
De Ruijterkade 143
1011AC Amsterdam
info@container-solutions.com (mailto:info@container-solutions.com)

(http://www.twitter.com/@containersoluti)

(http://www.linkedin.com/company/container-solutions) (mailto:info@container-solutions.com)

Sign up for our newsletter