FAQ

DOCKER

The ultimate deployment tool, or just another tech fad?

BEN EVERARD

Q Ok, let's start with an easy one: what is Docker?

A That's simple: it's a tool set for managing deployments of containers.

Q You're stretching the definition of 'simple' a bit. Can you break it down for me a bit more? Let's start by explaining what a container is.

A OK. Remember that Linux is just the kernel and the operating system is this plus all the tools that sit on top of it?

Q Yes, but I thought I got to ask the questions?

A Sorry. A mild digression only. The Linux kernel is the bit that sits on top of the hardware and controls access to the CPU, memory, and all the other stuff that makes up your computer. Normally, you can only use one kernel on a computer at a time. However, you can have many copies of everything else. Containers are a way of encapsulating this 'everything else' so that they can share a kernel, and this enables you to run multiple distros on the same hardware.

“You could have a Docker image for OwnCloud and another for WordPress.”

Q Like having a dual-booting Linux system?

A No. Using containers, you can run multiple distros at the same time, or, as is more common, using the same distro multiple times.

Q Ah, so containers are a form of virtualisation, like Virtualbox or Qemu?

A From a user's perspective they're pretty similar. You have a host OS, and within that host OS, you can boot more versions of Linux. However, at a technical level, they work in very different ways. In virtualisation, you have an application on the host OS that simulates a CPU, then you have another entire OS (including kernel) that runs on this simulated CPU.

In containers, you only ever have one kernel. It's the same for the host operating system and the other operating systems that you run. The containers have their own chunk of the filesystem where they keep all their data, and behave exactly like independent OSes, but it all runs atop the same kernel.

There are advantages and disadvantages to this. Because they run the same kernel, you can't run different operating systems like you can with virtualisation. However, on the other hand, because they don't simulate the CPU, the performance is better.

Q Great! Now I understand it, and we've still got a page to go. Shall we just stick a picture of Linus Torvalds in there and nip off to the pub early?

A Not so fast! That's containers. We need to get onto Docker itself.

Q Oh right, yes. You said before that it's a tool set for managing deployments on containers. I know what containers are, but why would you want to deploy them anywhere?

A The big advantage of containers is that you can encapsulate an entire environment into a single block. This enables developers to pull all the libraries, data, and software into a single container and distribute this. By sending the container, rather than just the software, it means they don't need to worry about dependencies, different configurations, or anything like that.

Q So it's a bit like statically compiling software, but including the whole OS?

A I'd never thought of it like that, but I suppose it is really.

Q Sounds awesome! What's the inevitable downside?

A Nothing major, but the containers will take up more disk space than just the plain software, and they have to be updated separately to keep them current with the latest bugfixes and security patches.

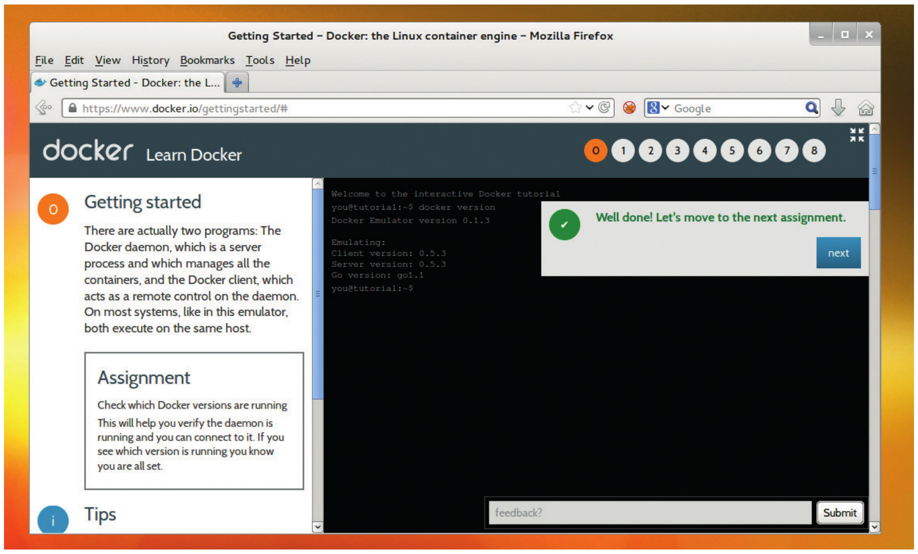
Q So is Docker poised to become a universal replacement for apt-get, Yum, and all the other package managers?

A Not really. No one's suggesting that containers are a sensible way of installing all your regular software. The main target market for Docker is people providing services across a network. For example, you could have a Docker image for OwnCloud, and another for WordPress. They would each have their own environments with everything installed, set up, and ready to run.

By keeping everything contained in this way, it's really easy to customise and deploy. A developer could pull the Docker image to his or her development machine, make any changes they like, then push it to the server. They don't need to worry about the development environment being different to the live environment, because the whole environment is included in the container. It it doesn't matter if it's developed on bleeding-edge Arch Linux, and deployed on ultra-stable Centos, it will always run the same. As well as making it easy to develop, this should remove much of the hassle of setting up test servers, or migrating to a new environment.

The developer can also make any changes they need to the environment without worrying about how these may affect other software running on the server, because that software will be in a separate container.

Q I almost understand your first point now: 'a tool set for managing deployments of containers'. What sort of tools are typically in the set?


Getting Started – Docker: the Linux container engine – Mozilla Firefox
File Edit View History Bookmarks Tools Help
Getting Started - Docker: the L...
https://www.docker.io/gettingstarted/#
Google
docker Learn Docker
0 1 2 3 4 5 6 7 8
next
feedback?
Submit

Getting started

There are actually two programs: The Docker daemon, which is a server process and which manages all the containers, and the Docker client, which acts as a remote control on the daemon. On most systems, like in this emulator, both execute on the same host.

Assignment

Check which Docker versions are running

This will help you verify the daemon is running and you can connect to it. If you see which version is running you know you are all set.

Tips

```
Welcome to the interactive Docker tutorial
you@tutorial:~$ docker version
Docker Emulator version 0.1.3

Emulating:
Client version: 0.5.3
Server version: 0.5.3
Go version: go1.1
you@tutorial:~$
```

Well done! Let's move to the next assignment.

You can get a taste for Docker without the hassle of installing anything by trying out the project's interactive tutorial at www.docker.io/gettingstarted.

A The main tool is (unsurprisingly) called **docker**, and it has options for getting and manipulating containers. There's a repository of containers that have been made for common purposes. You can grab these with:

```
docker pull <name>
```

Then, once you've got one installed, you can run commands on it with:

```
docker run <image-name> <command>
```

That's the basic use. There are also a few options to help you manage the containers. It's complex technology, but it's surprisingly easy to use.

Q This all sounds so good, you must have found it really useful when setting up the Linux Voice web services.

A Actually, no. We didn't use Docker. It is, as you say, really good, but it's also really new and still under heavy development. The current version is 0.8, and the people behind it recommend that you don't use it for production systems until version 1.0 at least. It's not unstable, but it's not yet as mature as we like our server software to be. Of course, some people are using it on production machines. We're just a little conservative about such things.

Q Ah... so I should expect it about the same time GNU/Hurd is ready? Perhaps in time for Linux Voice #100 (or Hurd Voice #1).

A Less of your cheek! The first release was in March 2013, so it's just one year old (happy Birthday Docker!). In that time it's come all the way to version 0.9. The earliest plan we heard about was for a release of version 1.0 in October 2013, but by November that year, it had been pushed back to February 2014. At the time of writing, there was still no sign of it, but we don't expect it to be much longer. Of course, just because it's called version 1.0 and the team behind it say that it's production-ready doesn't mean it's ready for everyone. Sysadmins are a conservative species by nature, so we don't expect many people to start using it in important services for a while yet.

Q I'm not a conservative sysadmin, I'm a reckless maverick programmer. How can I get started with Docker?

A You won't find it in many distros' repositories just yet, but there are packages for it for most major Linux distributions at **http://docs.docker.io/en/latest/installation**.

Q You said it ran on Linux containers. I have this friend who runs a commercial OS and won't listen to reason. Can he run it?

A Sort of. You can run Linux on OS X or Windows in VirtualBox, then run Docker in this. There are instructions at the installation website above. Of course, it's far better just to give your friend a talking to about the advantages of open source systems.