DOCKER TUTORIAL FOR BEGINNERS

Learn Programming, Containers, Data Structures, Software engineering, and Coding

Andrew Lee

Docker Tutorial for Beginners: Learn Programming, Containers, Data Structures, Software Engineering, and Coding

Andrew Lee

Published by MC Publishing, 2019.

Docker Tutorial for Beginners

Learn Programming, Containers, Data Structures, Software Engineering, and Coding

Table of Contents

_____

Chapter 12: Image Build Strategies

The Docker Build Strategy

Source to Image or S2I Build Strategy

Custom Build strategy

Pipeline Build

Chapter 13: Logging Strategies That Work the Best in Docker

Strategies for Docker Logging

Chapter 14: Best Practices for Using the Docker System

Instructions for the Dockerfile

Chapter 15: Is the Docker Cloud Important and More About Setting Up Your Docker Cloud Account

Introduction

_____

Congratulations on downloading Docker Tutorial for Beginners and thank you for doing so.

The following chapters will discuss everything that you need to know to get started with Docker for your own needs. Docker is a computer program that helps you to turn your operating system into a virtualization tool, without needing to take up a lot of space or slow things down. Creating an organization and working with the containers and all the other parts that come with this system can be important in helping your business run smoothly.

Inside this guidebook, we will talk about all of the different parts of the Docker system and the steps you can take to see results. From learning how to set up your Docker organization to knowing how to work with the nodes, the containers, and more, this guidebook will make sure that you are ready to go and use Docker as efficiently as possible.

Whether you have been wanting to try Docker out for some time, or you have just heard about it and think it is the best option for you, this guidebook can give you the answers that you need. Make sure to check out this guidebook to learn everything that you need to know about setting up and using the Docker computer program.

There are plenty of books on this subject on the market, thanks again for choosing this one! Every effort was made to ensure it is full of as much useful information as possible, please enjoy!

Chapter 1: How to Install Docker

_____

Docker is a type of computer program that is able to do virtualization at the operating system level. This is often known as containerization. It was released in 2013 and was developed by the company known as Docker Inc.

Docker is often used as a way to run software packages, which are known as container in this system. The containers are going to be isolated from one another and they will bundle their own tools, libraries, applications, and configuration files. They are able to communicate with each other through some channels that have been well-defined. All of these containers are going to be run through one single operating system kernel, which makes them more lightweight compared to the other virtual machines that you may have used.

These containers are going to be created by images. These images are going to help specify the precise contents that are found in each container and can consist of many different layers overall. The images that are used here are going to be created when we combine and modify standard images that are downloaded from a public repository and then used in this way.

Docker has been developed to be used primarily with Linux because it is able to use some of the isolation features of the kernels in Linux, such as the kernel namespaces in order

to help the independent containers to run within a single Linux instance. This can help to get rid of some of the overhead that usually comes with starting and maintaining a virtual machine.

You can use Docker in other systems as well. In this guidebook, we will take some time to look at the different ways that you can use the different components and even download them on Mac and Windows computers as well. But many of the codes that we use and the examples that show up are going to rely on the Linux and Ubuntu system.

To get the Docker software to work the way that you would like, there are three main components that are important. These are known as:

Software: The Docker daemon, which is known as dockerd, is going to be a process that is persistent and that is able to manage the Docker containers and handles the objects that are found in these containers. The daemon is going to listen for any of the requests that are sent through the Docker Engine API. The client program that comes with this, which is known as docker, is going to provide you with the command line interface so that you are able to more easily interact with the daemons of Docker.

Objects: The objects in Docker are going to be various entities that are going to be used to assemble any kind of application that you use in Docker. There are three times of objects in this language known as services, containers, and images.

A container is going to be a standardized and encapsulated environment that is responsible for running an application. It is going to be managed using the Docker CLI or API.

An image is going to be the read-only template that is used in order to build the containers. The images are going to store and ship the applications.

A service is going to allow the containers to be scaled through more than one Docker daemon. The result here is going to be known as a swarm, or a set of cooperating daemons that will be able to communicate through the API of the Docker system.

Registries: This is going to be the repository for the images in Docker. The clients are able to connect to the registries to download or pull the images to use or upload, or push these images that they already built. The registries can be either private or public. The two main public registries that you can use with this system are known as Docker Cloud and Docker Hub. Docker Hob is considered the default registry where Docker is going to look for images.

Now that we have reviewed a bit about what Docker is all about and how it works, let’s take a look at some of the steps that are needed to use this system by installing it to your computer.

How to Install the Docker System

The first thing that we need to take into account when we get started with Docker is how to install it onto our computers. It is impossible to actually use it if you don’t have it installed properly. You will be able to find Docker on Ubuntu and RHEL systems, and sometimes it can be used on other systems as well. Any time that you are installing Docker, you will be able to make it work so that it downloads automatically with the help of a droplet. However, there is a bit of script that is needed to ensure that this launches in the proper manner. Let’s take a look at some of the steps that you can take to launch Docker on your system properly.

Installing Docker on Ubuntu

First, we need to look at the steps that are needed to get Docker set up on Ubuntu. To make this easy, we want to work with a pre-built application. And we want to make sure that the system or version of Ubuntu you are working with is the 64 bit 14.04 VPS. The steps that you need to get this one done includes:

Make sure that you take a moment to update the droplet. You can open up the command prompt and type in the following

Sudo apt-get update

Sudo apt-get -y upgrade

Check to see that the aufs support is available for you to use. If not, turn it on.

To do this, you can use the following code “Sudo apt-get install linux-image-exra-‘uname -r’”

The repository key needs to be in the package and you must verify it before you go through the process of using it.

To add in the repository key, use the following code: “Sudo apt-key adv –keyserver hkp://pgp.mit.edu:80 –recv-keys 58118E89F3A912897C070ADBF76221572C52609D

Then you can add in the docker repository to the Apt Sources. You can then use the following code to make that happen:

echo “deb https://apt.dockerproject.org/repo ubuntu-trusty main” | sudo tee /etc/apt/sources.list.d/docker.list

You can then update the repository with the new addition using the following code:

Sudo apt-get update

And finally, you want to go through and download and install the Docker and all the information that goes with it. The code that you will need to make this happen includes:

Sudo apt-get install docker-engine.

The last thing that you need to work on here is to download Docker to the browser. This is important to ensure that it can be properly installed on your chosen operating system.

Now, the default firewall that comes with the Ubuntu system, known as Uncomplicated Firewall, is going to automatically deny all of the forwarding traffic. This is something that we need to fix in order to get Docker to work with our system. In order to enable forwarding on this system, we need to use the following steps:

First, edit the firewall configuration with the help of the nano text editor:

Sudo nano /etc/default/ufw

From here, scroll down and then find the line that starts with DEFAULTFORWARDPOLICY

When you find that line, you can replace the DEFAULT_FORWARD_POLICY= “DROP” with DEFAULT_FORWARD_POLICY= “ACCEPT”

From here, you can press on the CTRL+X. When it asks you to approve, you can push Y to save and close.

Now we need to reload the UFW. To do this, use the code:

Sudo ufw reload

Chapter 2: The Docker API

_____

Once you have the Docker program set up on the Ubuntu system, it is time to learn more about how to use this program and all the cool things that you are able to do with it. The remote API of Docker is going to have a schema model that you will find pretty easy to open. Whenever you are taking a look at this model, you will notice that the properties that are unknown at that time to the messages that come in are usually because they are being ignored. The user will need to take some time to deal with this to ensure that these unknowns aren’t going to break any of the daemons of Docker.

Docker is considered an open source platform that is used to build, deploy, and even manage the containers that are needed for various applications. This system is going to provide a lot of features that will allow the applications you create to run on any server, even under different systems than what it was created on and on many different platforms as well.

The API that comes with Docker will allow developers to access and even integrate all of the functionalities of Docker with any other applications that they want to use. Some of the methods that use the Docker API include managing user accounts, managing files, and even retrieving repositories and containers that have the associated data.

Setting the API up the proper way can make it much easier to work with this system. After we get it set up, take some time to look through the API and experiment with it a bit. This ensures that you learn all of the different functions that come with Docker and that you are able to get the most out of using this kind of program.

The Classic Approach to API

Before we can learn some more about how the API works in Docker, we first need to learn how classic API works and then we can compare. An API is going to be developed in these situations and it is going to issue a remote call to either a service or a server. These calls are going to be handled by the framework that is known as external to the API. With this framework, there is going to be a request for resources that are external to the API server in the form of the dependencies. The reason that this happens is because it allows for the code to function in the methodology that the designer created it for.

Even though this is the system that has been used for many years, it is often seen as antiquated in many different ways. It is going to fully depend on the developers to update the system, to maintain the right controls for the version, and to handle any vulnerability in security that may come along.

Plus, there is the issue of many of these dependencies being proprietary, or at the very least they are put in the hands of just one developer. This means that all the maintenance of the code is maintained outside of the API, and any of the changes that happen to functionality, modifications to any extra dependencies, and failures that show up in security could really cause a bit failure.

Barring all of these issues concerning dependency, the classic approach here is really resource heavy and it is really slow. It requires that the developers of the API host the entire system in a web of interlacing API’s in an attempt to hack together a system that actually functions well. It is a

delicate and functional ecosystem that is pretty neat in how complex it is, but with all of this complexity, there is the potential for the classic approach to basically be a house of cards. One little thing and the whole system can collapse.

The Docker Approach

Docker is a different kind of approach to the API that we just talked about before. Docker is going to allow for someone to remotely use the operating system images and the infrastructure that is there in a way that can help to distribute the dependencies, the functionalities of the system, and the core services inside the API itself. Docker calls these containers. These containers are like the virtual machine of the system, but much better.

A virtual machine, or VM, is able to package the applications with the libraries, binaries, operating system, and dependencies, and all the bloat that seems to come with all of these. This can be just fine when you are working with enterprise workstations and remote desktops, but it can lead to a lot of waste in bandwidth, and in truth, using it with APIs is a bad choice.

These containers, on the other hand, that come from Docker are going to be more self-sufficient compared to what we are able to see with the traditional API. They will contain not only the application, but also all of the dependencies that come with that application. It will also use what is known as a kernel that is communal to the other applications that show up in the user space of the operating system of the house. This is a good thing because it frees up the container, allowing it the opportunity to work on any system and it can remove the bloat found on the operating system as well.

More about the Docker API

The API is going to be the rest where you can find a few commands that are more complex. These may include some commands like pulling or attaching your connections to the HTTP.

The default is that the commands are going to be listed on the unix:///var/run/Docker.sock so that the consumer has to have the base access in order to interact with this program. In the event that you decide to name your group Docker later on, then the system will be able to apply the ownership of the group's socket in the right place to make things easier.

If you want to be able to connect to a daemon Docker, then you must make sure that the cURL is 7.40 or higher. If you use one that is in this range, then the socket flag will become available for you to use and it will run in the correct manner. Whenever this kind of cURL is run through the daemon, then the socket is going to default in order to use the different codes. Of course, you will only be able to do this if the right version is used, or if the local host is taken out of the address.

In the event that the Docker system becomes bound, then you will need to go through and use a different kind of socket path. The path for the socket and the port will need to reference the URL that you want to use. Depending on which version of the Docker system you choose to work with, a different type of API can be called. Should you go through and install an updated version at some point, then

you will see some of the newer properties returned to you, even if you happen to be using an older API.

Docker Machines with the API

Any machine that is based on this Docker program is going to have a daemon that it is responsible for hosting. This machine is going to use one of the sockets that have been specially encrypted for TCP, but it will also use the TLS. This is helpful to know because it means that a machine that is using Docker will not need to set up some extra conditions in order to get the functions of its cURL to get used any time that you test the requests of the API.

The containers that are found inside of Docker will need to have events that are going to happen but will not affect the container that it is in. Some of the events that you will see inside your containers in this program include:

Exec detach: This is when the client is going to be detached from the process of execution.

Export: This is going to be an event where the export in Docker is going to be emitted.

Detach: This is an event that is going to happen any time the client has moved away from the container process.

Exec create: This is an event that is going to happen when the Docker exec function happens.

Exec start: This is when the Docker exec is going to be emitted, but it only happens after the function that you created inside of Docker has been used.

The RMI function of Docker is going to go through and get rid of any events that are untagged if the images name is removed. This helps to keep things organized inside of the system. The RMI is also going to take some time to delete any events when the image has been deleted by the ID for that image.

Chapter 3: Nodes and How They Work in Docker

_____

The next thing that we need to take a look at is the Nodes that show up in Docker. There are several types of nodes in this system and each of them will work together to ensure that the different tasks and activities for each application are done in the correct manner. Let’s take a look at the nodes that you can find in Docker and how they work!

The Two Main Types of Nodes

In Docker, there are going to be two main types of nodes. These are going to include the manager nodes and the worker nodes. Let’s take a look at how each of these work and how you can use them to make Docker more efficient for your needs.

Manager Nodes

Manager nodes are going to be in charge of handling any of the cluster management tasks. This means that they can help out with things like maintaining the chosen cluster state, doing all of the scheduled services, and using swarm mode HTTP API endpoints.

With the help of implementation from Raft, the managers are able to maintain an internal state that is consistent for the entire swarm, and all of the different services that you have set up to run on it. When you are trying to test all of this, it is fine to run the swarm with just one single manager. If that manager is inside a single-manager swarm and it ends up failing, the services on that will still run, but you do need to recover by creating a new cluster to handle it.

To help you take advantage of the fault tolerance features that come with the swarm mode, Docker often recommends that you go through and implement an odd number of nodes. This should be determined based on the high-availability requirements of your company. If you are working with multiple managers at the same time, it is possible for you to recover from any failure of the manager

node, without having to worry about some downtime. Some things to remember about this include:

A swarm that has three managers is able to tolerate a maximum loss of one manager.

If you are working with a swarm that has given managers, then this means that the system is able to tolerate a maximum loss of two manager nodes at the same time.

An “N” manager cluster is going to tolerate, at the most, a loss of (N-1)/2 managers.

In many cases, Docker recommends that you stick with seven manager nodes for each swarm. This provides you with some protection in case one of them fails, but keeps you from running into too much trouble with bloat in the system or other troubles.

Remember that just because you add in more managers doesn’t mean that there is a higher amount of performance or more scalability in the system. This is a mistake that many companies will make when they start with Docker. In fact, the opposite is true. If you start adding in too many managers to the system, it can cause a lot of security issues, slows down the response time, and more.

Worker Nodes

Now that we have taken a look at the manager nodes, we also need to focus a bit on the worker nodes. Worker nodes are instances in the Docker system that has the whole purpose to execute the containers that are there. The

worker nodes aren’t going to take part in the Raft distributed state, they won’t help with making any of the scheduling decisions, and they won’t serve the swarm mode HTTP API either.

You can create a swarm of one manager node, but you can never have a worker node without having one or more manager node present. By the default in this system, all the managers are also going to be workers. In a cluster that just has one single manager node, you are able to run commands, such as “docker service create” and the scheduler will place all of the tasks that you have on the local Engine at that time.

To make sure that the scheduler isn’t able to place tasks on the manager node in the multi-node swarm, you need to change the availability of that particular manager node. Make sure that it is set to “Drain”. The scheduler will then be able to stop tasks on any node that has been turned on to “Drain” mode and will schedule all of the tasks that need to be done to those that are on the “active” node.

Changing the roles

It is possible for some of the nodes to change as needed. For example, the manager nodes are all considered worker nodes as well. But it is possible for you to promote a worker node to a manager node with a simple code. If you would like to do that, simply type in the code “docker node promote.”

One time that you may want to promote one of the worker nodes is when you take one of the manager nodes offline to perform maintenance. This ensures that there is still a manager node in play doing the work and will make the system run smoothly. Once you bring the manager node

back online, you can also demote a manager node back down to a worker node.

As you work on Docker, you will start to notice these nodes more and more. Make sure that you keep the swarm of manager nodes to a minimum to ensure that they don’t overload and bloat up the system more than it needs to be. Sticking with more than one can be nice, though as well because it ensures that you are actually going to be able to fix things if one of them fails. The Docker system will usually stick with about seven of these manager nodes to help you get the work done without having to worry so much about bloating up the system.

The worker nodes are important as well. They ensure that you are able to get the tasks done. You can choose which worker nodes are going to take on which tasks to ensure the most efficiency in the process. When the worker node and the manager node work together, the process is going to result in the Docker system doing the job that it should.

Chapter 4: Docker Images

_____

Now it is time to look at how we can build up images in Docker. There are a few tools that you want to use to make these images and they will include:

Tooling and CLI

Resource management

Process isolation

Security

Resource management, along with the idea of process isolation, is going to come from Linux itself, along with some of the other security measures. Some of the other security measures that you may want to use with the Docker images will be found on SELinux, which is a part that is separate from Docker. Along with these Linux tools, you may want to bring out a few others to help build images. The layering tool and the image content management tool are the most important for this.

The layered technique can be nice when it comes to images and content that you are working on in order to get an

abstract feeling built into any container that is responsible for a specific application. The image can then provide you with the foundation that is needed in that particular container if you do it properly.

From here, you can even take it a bit further. You can add in some additional layers to the foundation with the help of additional applications and tools. Containers are sometimes combined in order to create new images or alter some images that you already have. You can mess with the containers in this way with a function that is known as DockerFile. It is another way for you to create some foundational images that you can then add different parts to as needed.

To get started with the images, the first thing to consider doing is setting up the part that is known as “from instruction” You want this to be pointing at the image registry where you would like to pull that base image from. This can be from any kind of registry on the system that you would like, but usually it is common practice to have it in the Docker.ip registry.

After you have taken the time to create that initial layer, you can then add in some additional layers. Any layer that you add to it is going to be generated from the various directives that are found inside specific files. The run function is the one that you will want to use in order to provide running commands for that image you are working on.

As you go through this process, there are often other packages that you can install while using the package installation tool. If you do download a few of these and you are working with either Red Hat Enterprise or Fedora, then the tool is going to be named Yum. Further scripts and

additional content can then be added to the specific layers using the add function from the specific directory or URL that you want to work with.

After you have had some time to add all of the desired layers on your foundation, it is time to make the application more specified by creating an image that is usable more than once. There are two methods that you can use to do this. To figure out which method is the right one for you, there are some specifics to consider. First, you may want to take into account the type of content for your website that you will deal with. Then consider the nature of your image and the server online that you plan to work with.

Making a DockerFile that you are able to build on is the most common method that designers are going to work with. To do this, you will want to build up a DockerFile that is going to use your Fedora image, the one that was installed from the Apache package before the content that you need can be added. The good news is that the whole website can be made with one build.

Another option to use is the interactive method. This one is going to require that you use Yum and from there, you would need to go through and launch a shell for Bash. When this is done, you are able to utilize the CLI in order to lay down the primary image to the web server before you create an image that is reusable.

The method that you use is going to depend on your personal preferences. Some people like to go with the interactive method because they see it as an easier choice, one that gives them some more freedom along the way. But the DockerFile method is another option because it is usually seen as more stable and can help maintain the integrity that you get with your files too.

How to Create a Base Image

Now that we have spent some time talking about images in Docker and how they work, it is time to work on creating some of our images. First, we need to create the base image that we want to use. Most Dockerfiles are going to start from a single image that is known as the parent image. If you would like to have complete control over all the contents that you have in the image, then you may want to create your own base image instead of using the base image. The differences between the base image and the parent image are as follows:

A parent image is going to be the image that you base yours on. It is going to refer to the contents of the FROM directive in the Dockerfile. Each new declaration that occurs in the Dockerfile will make modifications to the parent image. Most Dockerfiles are going to start out with a parent image, rather than a base image. However, there are times when the two are going to be used interchangeably.

A base image either has no FROM line in the Dockerfile, or it has FROM scratch in it.

There are a few different ways that you can create your own base image if this is what you choose to do. We are going to take a look at one of the methods and provide you a good code to make this happen.

In general, we are going to start out with a working machine that is already running the distribution that you want to be able to package as the parent image. Some tools don't require this, but it is often going to make the process easier. It is also possible for you to also go through and build up an Ubuntu image. When you are ready to create a parent image with Ubuntu, simply use the code below to help you out:

$ sudo debootstrap xenial xenial > /dev/null

$ sudo tar -C xenial -c . | docker import – xenial

A29c15f1bf7a

$ docker run xenial cat /etc/lsb-release

DISTRIB_ID=Ubuntu

DISTRIB_RELEASE=16.04

DISTRIB-CODENAME=xenial

DISTRIB_DESCRIPTION="Ubuntu 16.04 LTS"

How Do I Manage Shared Images?

Once you have gone through and created the image that you want, you can then take some time to share it with any other Docker user. This is a nice thing because it means that it isn't necessary to store that image locally. Images that you can share will then be modified and reused in different ways depending on who you share them with.

In order to share your image, the first thing that we want to do is upload it over to the Docker hub. You can find this hub at Hub.Docker.com. This is a public registry that you can use, thanks to the creators of Docker maintaining it to keep images usable for the long term. Right now, there are more than 15,000 different images placed on the hub, which makes it a great destination if you aren't sure what images you need to have when you are creating a container.

The hub can also be useful because it makes it possible for you to work with a few different tools, ones that are great for improving the authenticity, privacy, and even the workflow of all the images that you create. This happens regardless of whether you wish to share this finished product or not. You may also choose to store some of your Docker images on GitHub if you would like.

When you are ready to access the Docker hub, the first thing that you can do is go onto the website (we listed it above) and then create your own account. The credentials that you create will get stored in your own Docker folder and saved on the server so that the Hub is easily able to remember you next time. You can also get started on this by

starting up the Docker prompt and entering in the code below:

Syntax: $ Docker Login

Once you have been able to successfully log in, the resulting interface helps you to search for repositories and you can push and pull them as needed. The commands will reflect what you want to do on the hub and you can find them in the CLI of Docker. As you search through this, you can look through and find the images based on their descriptions, who created them, and their names. This makes it easier to find exactly what you are looking for.

Pull or push commands are often going to be used in order to send an image to the hub, or to pull an image off the hub if you wish to use it. As you can guess by the name, the public repository is open for anyone who would like to use it, even if they don’t have an account. But the push function is only available to those who have an account. If you would like to be able to push and pull on the Docker system, then you must first go through the steps to set up your own account and get your Docker system linked up with it.

Chapter 5: Docker Organizations

_____

Docker is meant for many teams and organizations to use them. This can help to streamline the process, helps you to work better with your clients and customers, and can keep important information safe and secure. Being able to set your organization or team up inside the Docker system can be really important and can make a big difference in how well you are able to utilize the system. Some of the things that you need to know about Docker organizations include:

Organizations and Teams in the Cloud

You are able to create an organization in the Docker-based cloud to ensure that everyone in that organization, no matter where they are located, is able to share the same infrastructure and repositories across applications. A member of the organization, as long as they have the proper credentials, will be able to see what the team is working on based on what authorization they have within the team. Those who are considered admin will have more power along the way, and those who are just workers will be able to see just what has been set up for them. This ensures that everyone is on the same track and is kept up to date on the information that they need.

The admin members of the team who should be determined ahead of time are able to go in and start an organization. This also means that they are able to edit everything when

it comes to the team, from adding and editing the memberships of others and more. Those who aren't in the organization and those who haven't been added are not able to see any of the information that is inside of it. This includes the details about the team as a whole and the individual members who are on the team.

In Docker, the organizations are going to be broken down into teams. And then each of these teams is broken down even more into members. Users can't be added directly into the organization. From here, each organization is going to have some repositories, applications, and infrastructure that are associated just with them. Depending on the features that you choose to use with your team or organization, it can cost a bit of money to use this, but it is also possible that your team will be able to get into one of these organizations without paying any money at all.

There are five steps that come with creating one of these organizations. The first one is to log into the cloud that you will use. Once you have logged into the Docker Hub that we talked about before, you will then be given the option to create one of these organizations. All you have to do is click on the icon that is on the top right of the page. Once you have started the creation process for your organization, you will then need to enter the name that you would like to call your organization.

After you name your organization, you can then enter in the billing information. You will need to add this information even if you have no intention of using the paid services. From here, save the organization. The cloud will then get to work saving the information and then will move you into the organization view. You can then have the private account show up instead.

Once you have been able to create the organization, the relevant account will then be added to the owner of the organization’s team for the purpose of management. The first team that you work to create must have at least one other member in addition to you. You can always go in and add more or remove people at a later time if needed.

If you are inclined, it is possible to convert a personal account over to an organizational one instead of making a separate organization. Depending on what your accounts are for, this is a great option. However, if you do decide to go down with this route, you will not be able to log in with just the original email address. Anything that you link to the account is severed and all of the collaborators that are on there are removed as well. The good news is that any automated builds you associated and linked to that account will migrate over as well. Remember, once you move a personal over to an organizational account, you can’t undo the process.

You will also need to work with a valid ID for the account as the username will be listed just under the first member of the first team. Any automated builds that are relevant at this time are going to convert based on the settings in the organization to ensure that all of the current and the future members of the team have access to them.

It is easy to convert your account. First, you should get yourself logged into the account that you want to convert. Once you are logged in, you can choose the option for settings. Looking through this menu, you want to choose the option that is going to convert you to organization and then when the warning shows up, agree to it. Once the warnings are gone, you can enter in the Docker hub ID for the second member you want to add to the group along with you. Then select to save and continue. Your system is going to refresh

and you can then just use the ID you had on your personal account to log onto this.

Configuring the Owner's Team

For each organization you have, there is going to be a primary team that will be called the owner's team. The members who are added to this particular team will have more control over the organization and they will be able to manage the settings for that organization. This gives them the power to create, delete, view, and even change the repositories, nodes, and services found in the organization. The members of this team can also configure the settings, make changes to or delete the teams, alter who can be on the organization, and make any necessary changes to billing information.

Because of all the power and responsibilities that can come with this account, be careful about how you add members to this particular team. If you pick the wrong person, they can do some damage to the organization. You can add in some new members to the team as long as there is always a minimum of two members on the team. If you ever wish to leave the team and you were one of the people who started the team, then you will need to go through the steps of transferring ownership first.

To add members to the owner's team, you first need to log in using the credentials of the organization. You can then look for the option to add to the team; it will be located on the lower left-hand corner of the screen. From here, you can select the owners of that organization, followed by the option to add users. When you are on this screen, you can enter the ID of the user you want to add, followed by the option to create this user. You can then go back through

these steps until you have added in all of the new members for that particular team.

If you would like to transfer ownership of the organization, you can go onto the menu for the team and then click on the ID that corresponds with the future owner. You will see a menu and from there, you will be able to pick out the option to transfer ownership. Once you transfer the ownership, you will then be able to remove yourself from the team when you are ready.

In addition to this main team that goes with the owner, it is possible to work with other teams and create them. This allows you to provide other users with access to only some applications and tasks, rather than all of them. The process to do this is similar to what you did to create the owner's team, but you will specify that you are working with a new team. Once you have been able to create a new team, you can then set the level of access that you would like the members of that team to have. Unlike the owner's team, the members of any other team can be added or taken away at will.

Team Permissions

If you are setting up some additional teams on your Docker system, then you must make sure to set the permissions that you would like each team to have. The owner team is going to have the most permissions and they will be able to do the most with the system. But the other teams will be given different permissions based on what works best for your organization. To set up the permissions like you want them for the other teams, you can use the following steps:

First, you need to head over to the team details and choose the option that talks about permissions. You can then select

the relevant level of access that you would like to grant to that team. You can click on one or more out of application permissions, infrastructure permission, and monitors permissions.

When this is done, you will then be able to go to the option that is there for repositories, which can grant a specific team access to one, or as many as you want, of the repositories. Once you select the level of access that you feel the most comfortable with, you can then add those permissions to the profile of the member and the changes will save automatically for you. These steps need to be repeated for each member on the team to ensure each one gets the permissions that you want.

In addition, if you would like to give access to a specific repository to everyone in the same organization, without having to go through and do it with each individual member, then you can go to the repository and set that to public instead of private. Be aware that when you do this, it means that everyone who is on the Docker hub, not just those in your organization, will have access to that repository.

We will discuss a bit more about how you are able to set up your Docker account in the cloud and why it is so important to make sure you go through and give the right permissions to the different members of the team to ensure they can look at, modify, and make changes to the documents and other forms that are on the account.

Chapter 6: Bringing Out the Docker Toolbox

_____

If you are working with one of the older versions that are out there for Docker, then it is important to learn how to use the Docker Toolbox so that you can really take advantage of all the features and neat things that Docker is able to do. Downloading this toolbox is pretty easy to do and the installation is going to launch Docker automatically when it is all done.

Once you download the toolbox, you will be able to look inside and find the Oracle VirtualBox. This is the Docker machine that will be responsible for running any of the commands, the preconfigured shells for the various command environments, the Docker engine that runs the commands, the GUI for Docker, and Compose that can take care of composing all of your commands. All of these are important to ensure that you get the most out of your Docker system.

You will find that there are a lot of different options when it comes to the Docker Toolbox and you can definitely benefit from adding it on. We are going to take some time to look at the different methods of adding this toolbox to your computer, whether you are using a Linux system, a MAC computer, or a Windows computer.

You can also take some time here to install some of the other programs and tools that you will need. Be careful with what you install though. There are a ton of plug-ins and add-ons that sound impressive. But if you add on too many things that you aren’t even going to use, then you are just bloating the system and going to end up with a computer and a system that isn’t able to work as fast or as efficiently as you would like.

Installing on a Linux Computer

Since the Docker system is designed to work with the Linux system, even though it is set up to work with other operating systems as well, you won’t have to do anything extra in order to get the Docker Toolbox to work well with it. You can just download the Docker system and this particular toolbox

If you would like to add in some more plug-ins and add-ons in order to get more out of the Docker system, you can do that at this time. But for the basic toolbox that we are looking at here, you should find it already on the Docker system. You can bring out the command line to test whether it is set up and to search for it any time that you would like to use it.

How to Install the Toolbox with Mac OS

Before we get started, it is important to note that the Docker system is automatically going to work with kernels that are specific to Linux. This means that you will need to utilize the machine command right in the beginning in order to create your own virtual Linux machine in order to host the Docker engine on a Mac computer.

To start with installing this on a Mac computer, you need to consider what version of the operating system that you are using. To make Docker work, you must have Mountain Lion or later running on your computer. To figure out what version is currently on your system, you will go to the option for About this Mac which is found on the main menu. You can look right below macOS to see this.

Once you have been able to confirm which operating system you are on, you can then go to the Docker Hub and download this particular toolbox. Then double click on the resulting package. This helps you to launch the installer automatically and then you are able to use the toolbox.

Because of how the default is set up, the toolbox is going to go through and install all of the binaries that are relevant into its own binary folder. This ensures that all of the users in your organization are going to have access to the information. When you go through the installation process of the toolbox, you will then be able to input the password you use for Docker hub in order to give permission for the download.

The download can take a little bit to complete, so be patient and wait until all of the parts are installed. Once that is done, you can go through and verify to make sure that everything has been installed in the proper way. The virtual machines that you went through and created should be set up already to hold onto the changes that you make between each use to save you trouble.

If you have used the toolbox for some time and decide that you no longer want to have it there and you want to remove the toolbox, take note that you are also going to remove all of your access to Docker in the process. Uninstalling the toolbox is also going to remove all of the virtual machines that are active and that you created in this particular version of Docker. This is true even if they are not stored exclusively on the machine. But if the machine is stored completely in the cloud, you can still access these through the online menu. To uninstall the toolbox, you can open up the menu for this program, click on the uninstall option, and then wait for it to finish.

How to Install the Toolbox on Windows

You may also want to run the Docker toolbox on a Windows computer as well. To do this, you must make sure that your system is at least at Windows 7 and you have a 64-bit operating system. Also, double check that virtualization is enabled on the computer that you are using. If you already have one of these virtual boxes running, you must go in and turn it off before you try to install the Docker toolbox. Once that is done, you can go through and download and install the toolbox.

To begin, you will need to go to the Docker hub and then download the file for the toolbox. You can then double click on the file in order to start the process for installation. When asked on the screen, make sure to give this program the right permission so that it can go through and install the proper way. You will know that this wasn't installed in the proper way if you go in and try to launch the program and it shows you a control box for the user account rather than the command prompt.

If you do decide to uninstall, you can do it the same way that we did on a Mac computer. Just go in and uninstall from the program. Make sure that you are ready to delete because the same issues with things not saving that you worked on.

There are several errors that can come up when you are trying to install the toolbox onto your Windows computer. One common issue is the NDIS6 error. This is going to

happen in most cases when you try to install this toolbox on an older system because the drivers aren’t compatible and this won’t work. You are able to get around this problem though, so you don’t have to go through and purchase a new computer for this. You simply need to install a virtual machine that you have. Then go through and uninstall the Toolbox before reinstalling it before, making sure to choose the option to install the virtual box along with the drivers as you do it.

The Docker toolbox is a great thing to add into your program. It ensures that you have all the tools and other features that you need from the Docker program, and makes it work better overall. Make sure to download it onto your system if you don’t already have it.

Chapter 7: Common Mistakes Found When Using Docker Containers

_____

If you have ever used Docker in the past, you will know just how useful the containers in this system can be. First, they are fast as well as lightweight, which means instead of consuming thousands of MBs or more, they would only allocate the memory that is absolutely necessary for their primary purpose and nothing more. Even when only using that small amount of memory, they were still able to load as quickly as an average process on Linux.

Add to all of this that the containers are considered immutable, which means that they work even if they don’t include folders, library versions, or configurations in addition to the primary application. You know that if you tested an image and it was acceptable, then you will be able to use it in the production environment, no matter what happens later on.

Due to all of the advantages that you are able to get with the containers in Docker, many people like to treat these containers like they were similar to any other virtual machine. This may seem like it works in the short term, over the long term, it is going to lead to a lot of issues because it means the designer isn’t using the containers in the proper way. It is important to note that the biggest difference between containers and a regular virtual machine is that the

containers are disposable. If you don’t treat them like this, it is going to lead to some big issue later.

If you are going to use the Docker system and you want to work with the Docker containers, you must make sure that you are using them in the proper manner. If you want to make sure that you are avoiding some common mistakes that can happen for beginners and experts alike when they are using these containers, you need to avoid the following:

Avoid using these containers in order to store data

Containers can easily be stopped, replaced, and sometimes even destroyed. Not only that, but they should be each of these on a regular basis. For example, an application that you are working on that is using version 1.0 of a container should be set up to easily replace that container with version 1.1 without having to deal with the issue of potentially losing some valuable data. A better option for the designer to work with is to use volume to store data because you would be able to count on it staying around for the long term.

If you learn how to take precautions like this from the start, it is easier to ensure that the application is going to write to the shared data store, rather than letting it get copied directly over to your container. If you copy over to the container, you could use all of that different information when it updates or gets replaced. But if you make sure that you are saving the data to the right place, you will be able to find it later on if needed.

Never ship the application in more than one piece

If you learn how to think of these Docker containers as virtual machines, then you can see why it is easy for some developers to deploy an application into an existing running

container. But this is never a good method to work with. In reality, when you work with a continuous delivery pipeline from quality assurance to production, then the application that you choose to send out should always be part of the image.

Now, this isn’t meant to say that when you deploy the application into a container that already exists and is running, then it is never the right choice. In fact, it can be useful when you are in the development phase which will require a lot of debugging and deployment. But this should be the exception, rather than the rule. You should go through and make alterations to your plan after production so that you can move the information out of the containers and not risk losing all of it.

Avoid images that are oversized

To keep it simple, the larger the image, the more difficult it is going to be to distribute it in an effective way. This gets even harder if you spread it out over potential hundreds of connected containers. Because of this, it is always the best idea to only install the libraries or files that are absolutely required in order to run the process or the application in question. In addition to this, make sure that you avoid the temptation to have excessive packages or running yum updates. Doing this can add quite a few new files to the layer of the image and can really increase the size of the image at the same time.

Another thing to remember is that any updates to the system can trigger some file changes when you are working on a new rpm package. While you won’t be able to change the existing layers of the image, you are able to reduce the load of the rpm cache just working with a single run command. And you are able to make some modifications to

the run command and add in a command to clear out the cache to help with this issue. Of course, even when you work with this edited command, make sure that you go through and manually clean up as much of the space as you can when you finish up with the image.

Avoid images that are single-layered

In order to be effective when using a layered file system, you will want to ensure that you create a unique base layer image for the system that you choose to use. You can then create a new layer that will be for the definition of the username and another one for the runtime installation. On top of these, you can add in another layer for the application and its configuration. This makes it easier for you to go through and recreate the image when you need to. It can also make it easier for your designers to go in and manage and distribute the image when it is time.

Don’t use a running container to create an image

What this means is that you don’t want to go in and use the command Docker in order to create an image. This may seem like one of the best methods to use, and it does help you to capture the image, but it runs into trouble because you won’t always be able to reproduce it. As the ease with which an image can be reproduced is one of the most lauded elements in Docker, which means that you want to avoid these issues as much as possible, especially with the example we listed above.

Any source to image approach, including using Dockerfile, is going to be 100 percent reproducible. This means that it is going to be the best choices to go with each time. As an added bonus, the Dockerfile is able to track your changes,

but you must make sure that you are storing the changes and the image in the right source control repository.

Make sure that you don’t use the wrong tags

While you should use some tags to help you navigate through the layered system in your containers, you need to make sure that you are using the right one. If you use the wrong tags, you will make this process much more complicated instead of less. Specifically, you want to avoid the latest tag because it will be the most likely to cause problems later on when you want to either retrieve or build up a new image based on that previous image. If you tagged it wrong and you use the wrong tag, it can cause issues with that new image. The applications won’t run the way that you want because the parent layer was replaced or became incompatible with what you are working on now.

So, when it comes to using this system, make sure that you avoid working with the latest tag. Tagging can help you to do some really cool things with the images and can make it easier to work with your parent image. But if you tag incorrectly, you are going to end up with a mess when you are trying to get a few images to work together.

Don’t put together multiple processes in one container

One of the benefits that you get for using a programming system that is container based is that all of the containers are going to work together fluidly to achieve results that are better than the sum total. The strength is then mitigated if you try to force more than one process into the same container.

It can be tempting to try and do this. You may assume that using the container to help out with one process is fast, so if

you have that container work on more than one process, it should be faster. But in reality, there isn't a benefit to this practice. In fact, piling the processes on top of each other into the same container can actually make it harder to manage, update, and retrieve logs on each separate process and with the container as a whole.

Don't try to rely on IP addresses

Each container you work within this system is going to come with its own IP address. On occasion, this IP address can change if you stop the container for a bit and then try to start it back up again. Using some of the environment variables that are out there, such as the image codes from Postgres can be a much better option than relying on these IP addresses.

If you rely on the IP address, you could be setting yourself up for some trouble. You may assume that you know what the IP address of the container is. But what if the container gets a new IP address? This is possible and can happen at any time without much notice with it. If you try to rely on that, you would end up getting the wrong container, and maybe even none at all.

Working with a Postgres code or another type of code will make it easier to find the containers that you want. It allows all the containers in the system to communicate better with each other via a microservice and the containers can also pass along the right port information and the right hostname when it is needed.

Containers can be a great thing to use when you are on the Docker system. But you must make sure that you are using them in the proper way. If you fall into any of the mistakes above, then it can be difficult to make the containers work

the way that you would like. But when you can use these containers in the proper way with all of your systems, you will find that it is easier than ever to make the containers work effectively for your needs.

Chapter 8: Data Volume Usage Tips

_____

The next thing that we need to discuss when it comes to working with the Docker system is the data volume usage. When it comes to utilizing Docker in an effective manner, there are going to be two main ways that most designers are going to work with. The first one is going to be through the use of data volume in general. And the second method is going to be through the use of data volume containers.

A data volume is going to be a special directory that has been designated within at least one container, but it can be through a few, in such a way that it is able to bypass the standard file system. Data volumes can be really useful when it comes to any data that has to be shared or data that is persistent over the long term.

Volumes are initialized automatically each time a new container comes into being. Any of the data that an image contains is added to this particular volume when it is initialized, though this does not occur when a new host directory is mounted. Once this data volume is initialized, you are able to share and reuse it any time that you need through the different containers.

The changes to this data volume are going to be made directly, but this isn’t going to include any instances where a specific container image needs to be updated afterwards. And finally, they are persistent even if the initial location

where they were found is no longer one that you can use. They are going to be independent of any of the containers. The only way that you are able to ensure that a volume is gone completely from the system is to actively seek it out in the system and then manually delete it.

Add a New Data Volume

If you would like to add in a brand new data volume to one specific container, you would use the -v flag along with either the Docker create or the Docker run commands. -v can be used more than one time in a row to help you mount in more than one volume of data at once. An example of what you can do in order to mount a single volume in a container that will then run in the web application is below:

$ docker run -d -P –name web -v / container name/

In order to help yourself find one of the volumes that you have already created, you will need to use a command that allows Docker to go through and inspect. The results that you will see from Docker will include all of the details that you want to search for on the container in question. This will include any volumes that it may or may not contain. The output you see would look like something below:

“Mounts” : [

{

“Name”: “fac362...80535”,

“Source: /var/lib/docker/volumes/fac362—-80535/_data”.

“Destination”: “/webapp”.

“Driver”: “local”.

“Mode”: “”.

“RW” : true,

“Propagation”: “”

}

]

...

The details that you should receive from this would include a source that is able to specify the location of your chosen host and a destination that will specify the volume location of the given container. Finally, the RW is going to show you if the volume you are looking for is considered writeable or not.

Mount a Host Directory that Already Has the Right Volume

You can choose to create some new and additional volumes through the use of the -v flag. Another option that you can work with is to use the Docker engine and then add this to the container of your choice. If you would like to do this and insert it into your own webapp name and location, you can work with the code below to help you do it:

$ docker run -d –P –name web -v /src/webapp:/webapp name and location

The results of this will make sure that the directory that you specify is mounted into the right location as you commanded. If it already exists as part of the image for the container, then this mount command is able to overlay the existing content without any need to remove it. But if the mount was removed at one point or another, then the existing content is going to be accessible to the organization again.

Keep in mind that when you are doing this, it is very crucial for a container -dir to always show a path and one that is absolute. The results will either be absolute, such as a dst/docs if you are working with Linux. Or it can be a C:\dst\docs if you are working on a Windows computer.

If you are going to generate one of these absolute paths, then the Docker is going to bind and then mount to the path that you specify. If you just go through and use a name, then Docker will choose to use that name that you choose in order to create a new volume with whatever designation

you chose to provide. When it comes to the values of designation, you must begin it with including either a number or a letter, and then include either numbers or letters and then add in a hyphen or period. If you are creating a path that is absolute, make sure that it starts out with a forward slash.

Place a Data Volume Using a Shared Storage Volume

Aside from mounting a host directory in the container that you want, there are some plugins for volume in Docker that are going to make it easier for you to utilize shared storage volume at the same time. This can be useful because the volumes that are shared aren’t dependent on a specific host and this means that the volume is available for all of the hosts, as long as access is openly available and everything has been created as well as stored in the correct manner.

The best way for you to ensure that you are doing this is to use the command for Docker run. The use of this special command will help you create named volumes rather than creating them via paths. The command that we will have below will help you to create a named volume with the name that you choose, using the volume drive that you decide to specify. From there, it is going to make sure that the volume is available in the location that you picked. The command that you need to make this happen includes the following:

*$ docker run -d -P *

*—volume-driver=local (or to another location you specify) *

*-v volume name:/webapp *

Labeling Your Volumes

There are different systems of labeling that you can use, including SELinux and they are going to require that you go through and use the proper label and then place it on the volume that you will then mount the content into the container. Without having the right label, it is possible that the security that you are using on the system could prevent the process that you wish to run from actually running. If the process isn't able to run, then you won't be able to use any of the content that you want to access. One thing to note here is that when you keep Docker in the default setting, it is not going to change up the labels that you set in the operating system, and you may need to go through and do that.

To make sure that you are able to alter the context in how the container is labeled, you will either need to use the suffix of :z or :Z to the volume that you are mounting. Both of these are going to work to help you change the label of your volumes in the manner that you want. "z" is going to explain that the container pair you picked will share the volume content. This is going to cause the system to label both with a shared content label and both of them will be able to write and read the content there.

On the other hand, you can work with "Z" as well. If you choose to go with this one, it is going to cause the content in question to not be shared. This ensures that the content stays as private as possible. You will only be able to add in the private volume to the current container that you are in, so make sure that you are in the right container when you do it.

What to Do When it is Time to Mount and Create a Container to Hold Volume Data

For a designer to move some of the persistent data between several locations, or to utilize the data that exists in non-persistent containers outside of the said container, then you need to go through the process of creating a Data Volume Container with a specific name, and then you can mount up the data that is inside. If you want to create a name contained with shareable data, then you will need to work with a postgress or training image to ensure that all of the containers come in with as many layers in common as possible. This will help you to save a significant amount of space on the computer over time. To do this, you can use the following code to make it happen:

$ docker create -v /directory –name you want to use/postgress /bin/true

From here, you will want to go through and put the flag –volumes-from in order to make sure that the information you place in that container ends up where you would like it to go.

If this image from postgress already contains the directory you are looking for, then the volumes that are mounted from that container will then go through and hide the files, ensuring that the postgress image is not able to access it. Because of this, only the files that are in the container you choose will be visible at this time. Of course, you can also go through and set several parameters at the same time in

order to provide the new container with access to data from several different containers.

Another ability that you have is to extend out any of the chains that you created in this process. The code that you need to make this happen is below:

$ docker run -d –name db3 –volumes-from db1/postgres

If you remove any of the locations of the volumes, including any of those that aren’t mounted in the proper way, then the volumes that were created won’t be deleted like we mentioned. Instead, if you want to be able to delete these volumes from the disk, you must do it through the rm-v command from their final location. Doing this will make it easier to make any updates or changes that you need, without having to worry about how the older versions are going to cause problems with your work.

You need to really pay attention to what is going on when you create these volumes. The Docker system is not going to let you know if you are discarding a container before you use the -v option to make this a permanent deletion. Assuming that you keep moving ahead and you delete a container before you take this step, it is possible that you will end up with some dangling volume, which is one that will not be referenced through a container that is existing.

Data volume usage can be an important part of the Docker system and making sure that it is set up the proper way, and that you are storing it and giving it the right privacy based on what you would like to do with this, can make a big difference in the results that you get. Follow the tips above, and you will be able to use these data volumes in the correct manner.

Chapter 9: How to Create Docker Versions of Existing Applications

_____

At some point, you may find an application, maybe one that you already use or one that you would like to use, but it isn't a part of the Docker system. You may want to switch it over to be compatible with the Docker system so that you can use it along with the other parts of the system that you enjoy. Luckily, there are ways that you are able to create docker versions out of other existing applications that are there and we are going to discuss some of the ways that you can do that in this chapter.

Dockerizing is the name that has been given to the process of changing an application to ensure that it is still able to work in the proper manner, even when you move it over to a container on Docker. Even though this is a process that is considered fairly straightforward in most cases, there are some problems that can come up and you need to be fully aware of them ahead of time. This allows you to actually deal with them properly when they arise.

The first and most common problem that can show up is when you are working on an application that will use environment variables that will also rely on some configuration files. The second is going to occur when you try to send application logs to either STDERR or STDOUT when the default are files that already exist in the file

system of the container. However, using the Dockerize tool can help to solve these two big issues.

Configuration

First, we are going to take a look at the problem with configuration. There are many different types of applications that are going to work with the configure files to help them figure out the best way to work. In addition, there are different environments for runtime that will work with different values for separate sections of the file. An example of this could be a database connection. This is going to provide details for the environment and is used during the development. But this would then be different from the environment that it is going to be added into during protection. With the same idea, the API keys, along with any details that are considered sensitive, would be different depending on the environment that you use.

There are several methods that you can use in order to correct these differences inside the containers you are using with Docker. First, a designer could choose to embed all of the details that are relevant to that environment into the image directly. Then they could move on to using the control environment variable to help them make a decision on which files are used as the image is used.

Another method that you can use is to take volumes that need to be mounted and then bind the configuration that will be used any time the image is used. You could also choose to generate wrapper scripts. These are able to modify the data of configuration using the sed tool to help change any variables that are in the environment.

Out of these choices, embedding the details of the environment is not seen as the best solution. This is

because it is usually easier if the environmental changes that you work with do not require the image to be completely rebuilt from the ground up. In addition, this solution is not as secure as some of the other options and it is possible that some sensitive data, including the APIs and the login credentials, would show up in the wrong places.

Using volumes like we talked about in the last chapter can be a better option because they will ensure that those security details and sensitive data are kept away from the image. But there is the issue of this choice making the deployment process more complicated overall. Instead, you need to coordinate the configuration file each time something ends up changing. And adding in some of the details that you need about the environment is not always as easy as it seems. Sometimes you can work with custom scripting or the sed command to do this, but it gets very repetitive because you must go through and do this each time.

If you do choose that this is the best route for you and you don’t mind all of the repetitiveness that can come from it, you will at least have the knowledge that you can generate an image that works well with the ecosystem that you are creating in Docker.

However, you will find that the Docker system is going to contain the log that you need to both STDOUT and STDERR, which makes it easier to integrate, monitor, and troubleshoot into the logging system that you have centralized. These logs are nice and easily accessible through the Docker logs and any of the logs that are related to your API calls.

There are also going to be a few other tools that are able to automatically pull the Docker logs and then can ship these

off to the location that you want. Despite this ease though, there are many applications that are prone to logging into more than one location and they do this as their default. There are ways that you can work around this, but it is very tedious when you have to do this with each of your chosen applications.

Dockerize

This is where the idea of dockerize is going to come into play. It is able to come in and help simplify this process in several ways. First, it is going to allow for the generation of configuration files that are based on templates along with the variables that relate to a specific container's current environment each time it starts. And second, it is able to track down the different log files before centralizing them back to the STDOUT and STDERR that we talked about before. And finally, this process is going to make it easier for you to start this process and to get it to run through the container of your choice.

Going through these processes can help you to turn any existing application and make it work through the Docker system. This can really prove to be helpful if you are working with a particular program or application and you don't want to create it from scratch, or you just want to be able to add it to the Docker system and see some great results in the process. Take some time to look through Docker and decide which of the different options and applications that you would like to add into the mix.

Chapter 10: Strategies for Distributing Containers

_____

The next thing that we need to focus on in this chapter is some of the strategies that you can use when you want to distribute your containers. When it comes to utilizing the tool that is known as the Docker Cloud, there are going to be a few different distribution strategies that you can deploy, especially when you are using the containers through several nodes. Each of the strategies that we are going to look at in this chapter is going to allow you a way to change how your services are distributed across more than one container when scaling.

The Docker Cloud is going to provide the user an opportunity to supply their own Linux host (which is known as bringing in your own host) that can act as a node. They can then use this node to help them deploy their containers. This is only going to be possible after you go through and install the Docker Cloud Agent, but then, once that has been installed, you will be able to manage remotely from the Docker Cloud. Keep in mind that you are able to install this agent on any system as long as it is 64-bit.

Installing this Docker Cloud Agent

To get started on this in the right manner, the first thing that we need to look at is the ports of 6783/tcp and 6783.udp and make sure that they are working and open up to the

host that we want. You can also do this with 2375/tcp and make sure that it is opened up the right way. The first two ports are going to allow your chosen node to join the network so that it can discover the proper services. Then the second port is going to allow the Docker cloud access to the Docker daemon while using the TLS authentication to stay secure.

Once this is done, you will then be able to log into your organizational Docker cloud and you can choose the option for the node dashboard. When you choose this, click on the option for “bring your own node”. You will be provided with a list to show which distributions that support Linux are currently available. You will see a command line show up there and you can just copy and then paste this line to use it. The command that is provided will also include a token. This is important because it makes it much easier for the agent to talk with the Docker cloud.

The easiest way for you to go through and execute the command that you were given is to first go through and copy it over to your clipboard, and then you can paste it over to the host for Linux. Once you have been able to do this, the command is able to download in a separate script, one that is able to configure the Docker Cloud agent after you install it. You can then register it with your Docker Cloud.

After all of this has happened, you must take the right steps to ensure that it all worked properly. You can do this by finding the newly created host on Linux. It is going to be there in the node dashboard portion of your cloud. If you see it there, that means that this particular node is ready and it will be able to accept the container deployments for you.

Deploy the tags

In order for you to be able to utilize this strategy, you must be able to work with the deployment tags. These are important for ensuring that specific services go to the exact node that you want and that they don't get lost in the mix. Services that are tagged in the proper manner are then only going to deploy to the nodes that match with each and every tag that you give them. If there is a situation were no nodes matches a service's specific deployment tag, then you will find that the Docker Cloud is going to send back an error.

One thing to note here is that a node can have some additional tags than what one of your specific services requires and it can still work. But it must have all of the relevant tags. So, if your service has four tags, and the node has five, as long as it also matches up with the four that the service has, then the service is going to head on over to the node and stick there.

Tags can be anything really. They may be specified on services, on a single node, or on node clusters. If the tag is then added to a node cluster because it matches up, then all of the members that are already found in the cluster are going to go through an automatic process of inheriting the tags that you specify as well to make sure they all match up together.

There are a few steps that you can use in order to add a tag to one of your node clusters or even to a node when it is time to launch it. First, you need to choose the option for the node clusters. This is found on the right side of the screen for the navigation menu. You can then choose the option that shows up for creating. Once you click on this, look for the label of "deploy tags" and enter in the tags that should be assigned over to the cluster. You must remember that automatically, the Docker system is going to assign four

tags. Finally, you will want to go through and choose the option to launch the cluster of nodes to finish this up.

Update the tag on a single node or node cluster that is existing

In some cases, you will want to update the tags that you have on an existing node or node cluster. You can just click on the tags that are below each of the nodes in order to change or edit them as it is needed. If you move your cursor under the node and you can't see any tags, you can then click on the tag icon when it appears. Don't forget to click save when you are done in order to keep the changes that you make with the tags.

Change the tags on one of your services that are existing

Tags can be added or removed from a service, even if it is one that is already running. You just need to go to the service and select the detail view. From here, you can then click on edit. Once you are ready to edit, you should be able to see the currently available tag list under the heading for deployment constraints.

Once you go through and make the changes that you would like to see here, make sure that you save those changes. Of course, make sure that after you go through and change the tags on that service, it is going to keep on running as it was until you go through and redeploy it. To redeploy the service, you will either want to terminate all of the containers that are already running the service and then do a restart of them. Or you can go through and scale that particular service until it runs at 0 nodes, and then let it scale back up. Once the new container is deployed, it is going to match up with the edits of the tags that you set up.

Deployment distribution strategies

When you are first creating a service, you are able to take a look at the deployment strategies and see which one is going to be the most effective either through the UI of your Docker Cloud or through the API or CLI. When you are using a stack file, you are also able to specify a specific deployment strategy to help you define your service stack. No matter which method you choose to use, the emptiest node is going to be the default strategy to keep things moving along and to ensure that you can get the work done efficiently.

The emptiest node is going to be the strategy that is used the most when it comes to balancing out the total load of all the services you are trying to run across each node that is getting worked on in the system. You will be able to deploy this strategy by using the command EMPTIEST_Node. Then the system will go through and deploy itself to the nodes based on their deploy tags. From there, it will pick out the nodes that have the fewest number of containers at that time and work with them.

You can also choose to go with the high availability setting. This is a useful setting if you would like to increase the overall level of service availability on the network. You can deploy it with the command of HIGH_AVAILABILIY. When this is used, it is able to deploy the containers to nodes based on the assigned tags and based on the nodes that have the fewest containers that are running the service at that time. As such, the containers are going to spread across all the nodes that have these relevant tags as equally as possible.

And the final method that you can use is the every node method. You can use this one with the command of EVERY_NODE. It is going to deploy one container in every

node, as long as it has the right tags. If a service is going with this strategy, then a new container is going to be deployed in each node that is created with the relevant tags. You must remember that this service is one that you can't manually scale at all. If you use volumes, then each container, along with each of the nodes, will have a different volume.

If a client service is using the every node strategy, and it is also linked to a service that is based on a server that also uses this strategy, then the containers can be linked and the information will be able to be shared between them on a one to one basis. But the client services aren't going to be linked to the server services automatically with other nodes.

Other tips for Docker

In the default state, Docker is only going to run when you use the studio or through root. However, you may find that as you use the system, you will find some users who have a need to run Docker, but who are not able to get on and have access to the root users.

To help these individuals, you will want to go through and add them to a specific Docker group. This is a lot easier than it sounds and you simply need to work with the following command below:

-a USERNAME Docker

In this particular case, the username is going to tell you the user that you want to add. The group in question that you want to add them to should be one that exists. If it doesn't, this means that you need to take some time and create the group before you try to add someone to anything. Once you add the username, you can just restart the Docker program and the user in question will be able to access what they

need on the system without having to worry about the root user at all.

And remember to not underestimate the Dockerfiles. Deploying a new Docker container can sometimes become a hassle if you aren’t careful because many of the commands that are there can be tedious and unwieldy. Rather than trying to do it all in the command line, you should make it one of your habits on this system to use Dockerfiles, at least for some of the more common commands that you use.

A Dockerfile is basically going to be a short recipe of sorts that will describe the commands, the environment, and the files that are part of a chosen image. After you have been able to do this in a successful manner, you will then be able to access it and the Dockerfile will make it easier to access it through the command docker built -t docerfilename.

Chapter 11: Swarm Strategies in Docker

_____

Docker Swarm is going to be a kind of native clustering service. It is able to take a group of hosts through Docker that are unrelated and then turn them into one virtual Docker host that is effective and gets the job done. Docker Swarm is set up to work with the standard API on Docker, which means that any tool that can already communicate with the Docker daemon will be able to use the Swarm to scale different hosts. Tools that count in this will include Dokku, Docker Compose, Docker Machine, and Jenkins.

Like some of the other Docker projects, Docker Swarm is going to stick to a simple plug, swap, and play principle. This means that the goal of this is to get the program up and running as easily and quickly as possible. Because of this, it is easier for designers to swap the standard back end of Swarm with something else if you prefer and find it works better for your needs. This kind of swappable design is going to ensure an easy to use and even an out of the box experience with the program, while also making it easier for you to expand the production to a larger scale if you choose.

Before we go into some of the strategies that you can use in order to rank these nodes as effectively as possible and to increase the productivity that you can get from swarm, let’s take a minute to learn a bit more about Swarm and get a bit of a primer with this system.

What is Swarm?

Docker Swarm is seen as a scheduling and clustering tool that you can use on the containers in Docker. With Swarm, the IT administrators and even the developers on the team are able to establish as well as manage a cluster of nodes in Docker and they can do it as a single virtual system.

Swarm mode can also exist natively in the Docker Engine, even before you go through and download the whole system. It is a layer that occurs somewhere between the container images and the operating systems. This mode is going to integrate the orchestration capabilities of Docker Swarm into the Docker Engine of 1.12 and newer.

Clustering is a very important feature when it comes to Docker and to some other container technology. This is because it helps to create more of a group of systems that cooperating well, one that is able to provide some kind of redundancy. This is going to enable Docker Swarm failover if one or more of the nodes start to experience an outage. This kind of Swarm cluster is also going to provide the developers and administrators with the ability to either subtract or add in new iterations of the container, such as what is needed as the demands of computing change.

An IT administrator is going to be able to control the Swarm process and system through the Swarm manager. This allows them to orchestrate and schedule any of the containers that are there. The Swarm manager can allow users to create a primary manager instance and multiple replica instances as needed. In the swarm mode of the

Docker engine, the user is able to go through and deploy the manager and worker nodes right at runtime.

Docker Swarm Filters

When you are working with Swarm, you will notice that there are five filters that are used inside the system to help with scheduling containers. These five filters include:

Constraint: This is also known as a node tag. The constraints are going to be the key and value pairs that are associated with a particular node. A user is able to select a subset of nodes when they build up a new container, and then they can specify one or more key value pairs based on what they need to do in the system.

Affinity: To make sure that the containers are going to run on the same network node, this filter is going to tell one of the containers to run next to another container based on the label, image, or identifier that is there. This ensures that all of the containers stay together.

Port: When we are using this filter, the ports are going to represent a resource that is unique. When a container tries to open and run itself on a port that is occupied, it will be able to see this and will choose to move over to the next node that shows up in the cluster.

Dependency: When the different containers learn how to depend on each other, this filter is going to schedule them to work on the same node.

Health: If there is a time when one or more of the modes is not working in the proper manner. This health filter is going to make sure that, until you fix the node, there won’t be any containers that get scheduled over to it.

Creating Clusters on Swarm

When it comes to working on a new cluster in Swarm, one of the first things that you can do is head over to the Docker hub and pull out an image for this Docker Swarm add-on. Once you do this, you can then directly use the Docker program in order to do the necessary configurations for the Swarm manager. You can also spend this time working with the various nodes that are required so that this program works properly.

To do this, you can go to each node and open up its TCP port. This allows you to communicate with the manager of Swarm directly before ensuring that each node has Docker installed to it directly. Then, after this is done, you will need to go through and either manage or you can choose to create the different TLS certificates that are required to make sure the cluster is secured properly.

While the manual installation method is often seen as useful, it is not always easy unless you really know what you are doing with programming, specifically with this kind of programming. There is another way to do this, and a better choice for beginners, in terms of the Docker Machine.

You can easily install this Docker Machine to the Docker Swarm either through the existing cloud services that you have or through a data center that is localized. In addition, if you already have VirtualBox installed on your local system, you will be able to explore or build up the Docker Swarm on this local environment. This method can also generate, automatically for you, all of the security certificates that you need for any cluster that you create.

Before you go through and try to create a swarm through the Docker Machine program, you must make sure that each node you want to use is already associated with its discovery service. This service is going to associate a specific token with every instance of the Docker daemon that can run on the nodes. To make sure that you are able to list out all of the machines that are active right now on the system, you can try out the following command:

$ docker-machine ls

NAME ACTIVE DRIVER URL SWARM

*docker-vm * virtualbox Running tcp://192.168.99.100:2376*

When you have typed in this code, it is time to go through and generate the machine for Virtual Box. Make sure that you are labeling it in a local drive so you can find it later. The code that you can use to make this work for you includes the following:

$ docker-machine create -d virtualbox local

INFO[0000[Creating SSH key...

INFO[0000] Creating VirtualBox VM...

INFO{0005} Starting VirtualBox VM...

INFO[0005] Waiting for VM to start...

INFO[0050] “local” has been created and is now the active machine.

INFO[0050] To point the Docker client at it, run the following in the shell:

eval “$(docker-machine env local)”

Docker Swarm Schedular Strategies

After you have been able to go through and use the code above and some others to help you get the Swarm system all set up, it is time to learn a little bit more about some of the strategies that you can utilize to get the most out of this program. With Swarm up and running in the proper manner, and after making sure that it has been configured properly, the next item on the list to do is consider which strategies can be used in order to help with the ranking nodes.

The strategy that you pick for this is going to help Swarm to determine, in an accurate fashion, how the nodes are going to be ranked. This is then going to determine the best way to optimize the cluster so that it works as efficiently as possible for your needs. Any time that a new container is running for the first time after this, the Swarm program will be able to place it into the right node, the one that receives the highest of the rankings based on any strategy that you choose.

The strategy for ranking can easily be chosen by using the flag -strategy and then the strategy value with the command of swarm manage. Swarm is going to support a few values including spread, binpack, and random. Binpack and spread are some strategies that will automatically go through and compute the rank based on the nodes CPU and RAM at that time, along with how many containers it includes. But with the random strategy, it is just going to go through and pick out the new nodes to use at random, and it is often going to be used by those who are looking to debug all or part of the Swarm they are working with.

Any time that you choose to work with the spread strategy, the manager of Swarm is going to optimize the newest container to that node, to the one that has the lowest number of containers when things start. This is a great method to use if you would like all of the containers to be spread out thinly across a few different machines. When you do this, if any node goes offline for a bit, you are going to run into the risk of maybe losing some of your containers. If you do not pick out a different strategy in your commands, then the spread strategy is going to be the automatic choice for the Swarm system.

On the other hand, you can also work with the binpack strategy. This is going to optimize the system by adding new containers to the nodes that are set up with the most active containers already. This can be a good strategy to work with if you would like to be able to minimize any fragmentation in order to leave some more room for the larger containers that appear in the nodes that you don’t use as much. It can also help you to make due with fewer nodes in the first place because each of the nodes is going to be set up to pack in as much information as possible.

And the third option that you can go with is the scheduling containers strategy. When you are working with this method, it is important to remember that a container is going to occupy specific resources for the sum total of its life cycle, even if you have it set up inside the exited state.

The scheduling strategy is just going to check how many containers there are overall, regardless of which state they are in at the time. This means that if one of your nodes isn’t holding onto any active containers, it could still be bypassed if the node doesn’t have the most of these containers overall. To help you prevent this kind of scenario, you would either need to use the load spreading choice or you would

need to go through and remove any of the containers that have become inactive in the node that you choose to use for this.

Chapter 12: Image Build Strategies

_____

The next topic that we are going to take a look at is image build. When you are dealing with Docker, the term build is going to refer to the process of changing some simple parameters that have been inputted into the system into an actual image or another usable object. So when we want an example of this, a BuildConfig is going to be an object that can define the sum total of the building process. In some cases, it is going to be known more as a build configuration.

Each of these BuildConfig parts is going to be characterized by a build strategy and source material that comes from at least one location, but could possibly come from more. The strategy that you choose to use here is going to help determine the way that the process will work out, and each source is going to provide the right input to make it happen.

Currently, Docker is able to support four different types of build strategies and these are going to include Source to Image, also known as S2I, Pipeline, Docker, and custom. There are also six sources that are usually used in these scenarios and they include input secrets, external artifacts, Image, Git, Dockerfile, and binary.

Each of the strategies for building is then either going to choose to use or ignore the various types of sources along with determining which way they would like to use the rest. It is important to remember that some of these strategies

are not going to work well together. For example, Git and Binary don’t work well together while Dockerfile and Image are fine working on their own or together, and both of them work nicely with Binary and Git. Binary is also seen as a unique one because it is able to work just based on whether or not you worked with the OC start build.

The Docker Build Strategy

The first option we are going to look at for building images is the Docker build strategy. This one is based on Docker and is going to use the command Docker Build. What this means is that it only works if it has a Dockerfile repository that it can access, as well as all of the different artifacts that are needed to generate and make an image that is usable. In addition, it will need a context which is the file that will be located in a specified path or URL.

For this kind of build process, it is able to refer to any of the files that are in a specific context. For example, it could work with the instruction ADD to explain the file for referencing in the future. The URL parameter that you choose can then refer to one of three different resources. In this case, it is going to refer to a Git repository, a tarball that has already been prepackaged, and even a text file to keep things simple

If the URL does happen to be used and it points you to the Git repository, then this repository is going to be the context that you need for the build. The system is going to clone that repository in a fashion that is recursive, along with any of the submodules using the command git clone –depth 1 -recursive. The command is then able to run inside your temporary directory, which you will be able to find on the local host. As soon as this command can be successful, the directory is then going to be sent to the right daemon in Docker so that it knows the right context. Working with a local clone, such as what we described above, will make it easier for you to access repositories that are usually private,

based on the credentials that you have at your disposal. This includes the VPN’s.

Naturally, the Git URLs are going to be able to accept the context configurations in their section set asides for fragments that have a colon in place to separate them out. The initial part of the URL is going to specify the reference that Git will pull. This can either commit a branch, a tag, or an SHA. The remainder will then determine the subdirectory that can be pulled based on the repository that is used in the context of that particular build.

Source to Image or S2I Build Strategy

Now that we looked at the Docker build strategy, it is time to work on the S2I tool. This one can be useful when you would like to build up a formatted container of images that you can easily produce later on. This is going to give you results of an image that is ready to run and all you need to do is inject the source code for an application into the image container, before assembling the results into the brand new images that you want.

From here, those images that you just created and assembled will be able to incorporate the builder base image, as well as any relevant source information into the run command as soon as you are ready to use it. S2I is also going to be able to support additional builds, even those that are going to occur in an incremental fashion, and those that will reuse dependencies that were downloaded in the past.

There are many advantages in using the S2I system. The first advantage is the flexibility that comes with the system. The scripts with this can be written in a way that allows you to inject specific application code into a lot of different containers, including the ones that are formatted to work with Docker. This allows you to take advantage, as much as you want, of several pre-existing ecosystems. In addition, S2I is reliant on tar to help inject some source applications. This means that any image that is going to be used must be able to process any tarred content as well.

There are three elements that are needed for an S2I build. You will need the sources, the builder images, and the

scripts. Additionally, during the process, the S2I is going to generate a tar file as a means of containing the sources and the scripts that you need before it streams them into the file that will create your builder image. Before you try to execute on the script, S2I is going to untar this file and add in content to the location. This helps to determine the flag designation or the builder.

The scripts that are used for S2I can be written out with a few different types of programming languages, which makes it nice if you happen to know only one or so you aren't stuck trying to learn something new. A script that is able to describe the assemble process is something that is a requirement here. This script is going to build the required artifacts from a source and then can make sure they are placed in the right directories for the image.

Now, there is going to be a kind of workflow that is needed for this script. First, it is going to go through and restore any of the existing build artifacts that will help to support the new builds that you are doing. It can then place any application sources that you have in a specific location before it goes through and builds up the new artifacts for those applications. And then it is going to take those artifacts and install them into the locations where they are needed to make sure it all runs in the proper manner.

The run script will be needed here because this is the part that makes sure that the application is executed. You may also decide to work with the save-artifacts script because it can work to speed up this process by quite a bit. If you are working with Ruby, then it is going to be able to gather gems that were previously installed through the bundler, or if you choose to use Java, you will find that it gathers the .m2 content. Any relevant dependencies are then going to

be collected via the tar file before you can place them on the standard output.

Custom Build strategy

The next option is the custom build strategy. This is the one that will provide developers with a chance to generate a precise builder image and then this image is going to be the part that is responsible for the sum total of your building process. This is then going to allow for the maximum customization for the build process overall. A builder image that is custom is essentially a container image that has been formatted to be used with Docker, and that will contain an image that has a build process logic embedded inside it already.

This strategy can be used when you want to be able to fill in any gaps that are created as the popularity of one particular container starts to go up. If the build you are eying is going to be needed in order to generate some individual artifacts, including installable Zips, base images, packages, WARs, and JARS, then this is the best choice to go with.

In addition, this kind of strategy is going to make it easier for you to implement some additional extended build processes. Many of these are helpful when you want to do an integration or unit test along the way. The only limit that you have with the custom build strategy is your own personal imagination.

When you are trying to go through an image through custom builder, the first thing that will occur is that the image is going to pick up some different variables out of the environment, which can then provide it with the information that it needs. Without this information, the image is not going to be able to finish up with the build.

No matter what kind of variables you end up using here, it is best that you work on the custom builder image so that it is a read-only file. This ensures that it stays secure and that only the people who have access to these tasks are able to make any modifications. This information can then be parsed with the JSON build, rather than forcing you to have to rely on the individual environmental variables that other options will provide.

While you are given some freedom with the type of build image that you want to do with the custom strategy, you still need to create something that conforms to your organization and you must follow all of the standard build processes in order to get this to all work out well.

Pipeline Build

This build is going to be a strategy that makes it easier for any developer who is interested to define the Jenkins pipeline, and then you can execute this on the plugin for Jenkins pipeline. The build is going to commence at that time, and the OpenShift Origin program will monitor it (you can get this at OpenShift.org.) The workflow that comes for this particular strategy is then going to be defined by the Jenkins file and can be either directly embedded into the build configuration or sent over to the Git repository so that it can be referenced any time that it is needed.

When you define this through the pipeline strategy, the programs are going to instantiate a new server of Jenkins to successfully make sure that this pipeline is set up in a way that will execute and work properly. It is possible for you to use additional strategies of this kind for the same build order, and if you do, they will be able to go through the same kind of server to make sure that you get some uniformity in the work that you do.

However, be aware that once the Jenkins server is done with the work, it is not going to automatically delete like some of the others will. This is true even if all of the relevant configurations for the pipeline build have been deleted or removed in some way. The user needs to go through and manually delete this server to make it be gone.

Chapter 13: Logging Strategies That Work the Best in Docker

_____

Like with any other service type that you choose to work with, logging is going to be an important part of making sure that Docker is used in a successful way. Being able to analyze the logs correctly can help you to get more insight into a variety of things such as the performance, the reliability, and the stability of specific containers, and you can even learn more about the service of Docker as a whole. As Docker is a very flexible and dynamic program, there is still no single approach that is preferred when it comes to storing and gathering these log events. However, there are a few solutions that you can choose for logging in this program and each one will have their own negatives and positives.

Before you start to jump forward and pick out the logging strategy that you think suits you the best, there are a few questions that you should consider first. These questions include:

Consider how you would like to monitor and log the services on Docker to be the most effective. This could include things like how you will collect the data, any external and internal forces, and how you are going to monitor the infrastructure of the Docker.

How do you plan to utilize feedback in this system for the purpose of improving service quality and the management of the container?

Do you have any experience when we are talking about logging some less complex applications, or are you going to be a beginner starting from square one or the beginning?

If you do plan to start from the beginning, what is the easiest way for you to create a solution so that you are able to make the correct decisions with as little work in the process as possible?

While it is going to share a few similarities to traditional logging in other applications, when you do logging for a system like Docker that needs a bunch of containers, you will notice that it is a different thing altogether. When you are considering the strategies outlined below, there are a few different details that are important to remember and they include:

The nature of the containers. These containers are not permanent. They are able to come and go, start and stop, and can be destroyed and rebuilt on a regular basis. This means that if you want to store persistent log data in the containers themselves, this could be an issue. While volumes are able to store this kind of data on a persistent

basis, it is recommended that you export any of the data that you want to hang onto into a service that has been designed for this to keep the information safe.

The multi-tiered nature of containers: Keeping track of all the logs that occur in Docker isn’t as easy as you may think. Even running the container through a basic installation of Docker will mean that you end up with a minimum of three levels that you need to log. This would include the Docker container, the Docker service, and the host operating system. And the more complexity you add to the system, the more containers are running, and the more complicated this whole process becomes. What you need here is a reliable way to handle this logging.

The complex nature that comes with containers: Docker is seen as a complex coding method, but it has still been proven as the best choice for many enterprises. It does still have a few security issues that you need to address before it becomes a baseline standard. When it is compared to some of the other virtual machine containers, it can provide you with a broader vector for attack because they each contain the same kernel that your host uses.

Strategies for Docker Logging

There are a few different logging strategies that you can choose to work with when you are using Docker. Some of the most popular options will include:

Application logging

Logging via applications is the one that most developers are already familiar with. Basically, this is going to be the process where you will have an application that will run a specific container and then that container is going to take care of all the logging needs with the help of its own logging framework. An example of this is when you have an application in Java that will use the common log4j2 format and then it is going to send the logs over to another location, off the system, for collection and perusal when needed.

This process is then going to provide a migration path that is easy and intuitive if you are interested in logging the framework of any kind of application that is in existence already. The logs will then be sent out of the application out to a server that is remote and centralized, and it will also be able to bypass the Docker system, as well as the operating systems, completely. This is a good strategy because it provides the designer with a lot of control, but it does add in some extra load to the process of the application itself.

Application-based logging is the strategy that is going to resemble the monolithic logging that has been done in the past. It will allow developers a way to continue with the use of logging frameworks that have existed in the past, without

needing to add in some extra functionality to that host. The biggest issue with this though is that the framework for logging is going to be limited to the given container, which means that if the container does end up shutting down at some point, that filesystem is going to get lost as well.

In order to prevent this loss of important data, it is important to either go through and configure a data volume to hold that data, or you need to make sure that any logs you have can be forwarded to a remote destination, one that is more secure. There can also be some more difficulties when it comes to working on more than one container, especially when they are identical functionally because it is going to need some extra steps for identification for each container.

Logging with the data volumes

When you have a logging system that is based on the data volumes, you are then able to store the data in containers for the long term by using a specified directory located on the host machine. A single data volume is a part that can be shared with more than one container because this helps to centralize the process of logging through more than one service while still minimizing the additional load that may be required to make sure things work properly. However, using this method is going to make storing some of this log data difficult to move containers to a new host without some risk of losing the data for logging.

Despite this, data volumes are often seen as the best and most effective means to centralize and then store the logs you want for a longer period of time. This is due to the fact that they are going to automatically link to a specific directory that you can set up when you create it, which will reduce the chances of the data getting lost as long as these

containers stay in one place. As an added bonus, the relevant data is available if you ever need to access the logs, make copies, or generate the backup.

Using the Docker Logging Driver

This is another logging option that will work because it is going to forward all of the log events that happen in each of your containers over to a centralized location, usually found on the Docker hub server. The hub is then able to store them using the syslog instance that is going to run on the host. Also known as Loggly, this is a service that you are able to access by changing the logging driver of Docker to log into the system before it is able to work with the syslog tool. You are then able to finish off this circle by forcing this application to send the information to the container, the one that is created just to store the logs.

Once you take these steps, the container is going to receive the logs, rather than the operating system of the host. It also means that the container is going to be responsible for doing the job of forwarding the log event to the right place at the right time.

When we compare this to logging with the help of data volumes, Loggly is able to read the events of the log as they are happening, and it is going to do this right from the container's stderr output and the stdout. This can then allow you to centralize the container logs in a quick and effective manner with the help of the Docker service.

The benefit of doing this is that the containers are no longer going to need to read and then write off the log files. When they don't have to take the time to do all of this, it can improve the performance in most instances. In addition, the

log events are going to be stored in the syslog of the host machines and then we can access them with relative ease.

Using a logging container just for this test

With the last two methods of logging, we saw that there were quite a few advantages that show up. But they all share a disadvantage as well. This is mainly that they are going to rely solely on a service that must be able to run right from the host machine. But if you make your own dedicated logging container and use that, you will be able to manage the work of logging right from the environment that you created in Docker, no matter where you are, just as long as you can access this node.

Containers that are set up to just handle logging can then easily retrieve the logging events that show up from many containers, can collate this information, can aggregate it, store the information, and then forward it on as needed to the third parties. Of all the approaches and strategies that we have talked about, this is the one that works the best with the system that we have. It is then able to eliminate any dependencies that the containers are going to have when it comes to a host machine. And the best part is that it can do all of this without causing any issues to the capabilities of logging in the system.

Containers that are designed to handle logging can then manage a wide variety of logs, either for single or multiple containers, and you only need to add in a few changes to make this happen.

A dedicated logging container can work as a good choice for your logging needs if you would like to incompletely remove any dependencies to the host machine that could be detrimental. It can improve the amount of ease by which

you are able to move certain containers between the various hosts, something that can be a pain if you are using the data volume method.

In addition, it is going to let you scale the infrastructure for logging when you need to do so as adding in some new containers to this system can be as easy as noting the change in that logging container. You will also find that logging containers can easily retrieve many logs from different streams, including data volumes and stdout with minimal effort making them as flexible as some of the other logging solutions that we have talked about.

Using the Sidecar approach

And the final method that we will look at is the sidecar approach. When you work with this strategy, you will simply need to pair each of the containers you have with a second container that would be completely responsible for the logging needs of that first container. The first container would be the one that would run the application as you wish, and then it would be able to save all of the logs to the data volume or the second container that you set up. The secondary container is going to be free in order to use a process for file monitoring in order to tag all of the events that are seen as relevant, and then it will take these and send to Loggly. While it is similar to what we did with the logging container that was just for that task, the sidecar containers are going to provide a new benefit because they allow you to have some more transparency when it comes to where the log event originated from.

The major benefit of using this kind of logging is in how you are able to manage the events, specifically when we are talking about managing the logging needs the same way that you would go through and manage the applications.

Sidecar containers tend to scale a lot easier compared to some other methods, which can work better if your deployment situation is larger. This approach is also going to make it easier if you would like to incorporate more information when it is related to tracking specific logging containers with specific logged events. Since you can include some custom tags, you are better able to track when an event comes in, and which containers are providing you with the best tags on a regular basis.

It is a bit more difficult for you to set this strategy up, and sometimes it is seen as more complex. You need to be sure that each pair of containers is working properly, rather than just one, or you could lose some of the data that you need. If you are unsure about how well you can keep these up and running, another option would be the best choice.

Chapter 14: Best Practices for Using the Docker System

_____

Docker has the ability to build up some automatic images just by taking a look at the instructions that you place into the Dockerfile. This Dockerfile is going to be a text file that is able to hold onto each command, in the order that you want it to be executed, to ensure that your image is built in the proper manner. You can have the option of learning the basics of this by heading over to the website for Docker, and then heading over to the reference page. Should you be able to start with a new creation on this system, then this is where you will want to start things out.

Your container is going to be produced with the help of the image that the Dockerfile is able to define, and it must be as ephemeral as possible. What we mean by this one is that it must be stoppable and destroyable so that a new one can be built up and placed without needing a lot of configuration or setup in the process.

In many situations, you may find that it is best if you are able to find an empty directory and then place each of your Dockerfiles on there. After you have taken the time to do this, you can then take that directory and add it to any files that are needed to create the Dockerfile. To make sure that there is an increase in the performance of the build, make sure that you add in the Dockerignore file to this directory as well. The reason that you want this is because this

particular file is able to support the exclusion of patterns in a similar way that the .gitignore files are.

Now with any system, it is important to reduce some of the complexity that shows up in regards to file sizes, built times, and dependencies. To help with this, you must make sure that you never install any packages that are optional or that you won't need. If you know ahead of time that you need the package, then go ahead and install it to use. But if you are just installing something just in case, or because someone else recommended it, but you have no use for it at that time, it is just going to take up a lot of time and room on the system and can make it less efficient.

These are just a few of the best practices that you should follow when you are using the Docker system. Some of the other options that you can work with to get the most out of Docker include:

One container, one concern

When you are working on applications that use decoupling, they are found in several containers. This makes it a bit easier for you to scale as well as use the containers again at another time. For example, web applications need to work with three containers and each one is going to have its own image to manage the database of the web application. It will also work as an in-memory cache that can work in a decoupled manner.

If you have spent some time working with the Docker program, you may have discovered a method that requires you to stick with one process for each container. This is a good method, but it won't always stand true to say that you should have one single operating system in each of these containers. You will also find that the containers in Docker

will have the option of spawning extra processes on their own.

Let’s look at an example of this. We could have one container that is going to be named apple, but it also has the ability to go across other worker processes. Or you could have a container known as banana that is able to create a process that you need as requested. While it is usually a good method to have one process for each container, it is not something that you have to work with all of the time. You have to determine when it is best to use one container for each process, and when you can add more than one process to each of the containers. Use your own judgment to help you keep all of your containers modular and clean.

If you have one container that is going to depend on another container, then you must make sure that you use the network for Docker containers. This network will ensure that each container is able to communicate in the proper manner with the other.

Understanding the number of layers

We have spent some time talking about layers and why they are so important for making sure that your system works. The Docker system is going to rely on having lots of layers in the images and in the containers. These layers ensure that the system functions properly without needing to take up a lot of space or slowing down the system. But how do you figure out how many layers are necessary in the Docker system?

There needs to be a nice balance between the readability that you get with your images in Docker and minimizing the number of layers that you are using with each image. You want to have some caution, as well as some strategy when

it comes to the number of layers that you get to use in each container.

Multi-line arguments

When you are using the container with your Docker system, you want to make sure that you keep things as simple and easy as possible in case you need to go through later and make changes. There may be times when you need to go through and sort out some of your multiple lined arguments in a manner that is alphanumerical. You can also use this same idea in order to make sure you don’t make packages more than once, which ensures that the list is easier to update. Add to all this is the fact that it is going to make the PRs easier to read and review. And the nice thing about all of this is that you should add in a space before your backslash.

Building a cache

When you are in the process of building up an image, Docker is going to read through any of the instructions that you create and it can find in the Dockerfile. Each of these instructions will be done in the order that you write them out. It is going to read through the instructions just like we are used to reading through a list at the grocery store.

From this list of instructions, the system is going to read through each one and execute it in the order that it is written. First, each of these instructions is going to be examined in order to discover if there is an image that exists. This allows you the option to reuse it instead of creating a second duplicate image. Should you decide not to use the cache, then you need to insert the – no – cache = true command in the build.

If you decide to allow Docker to work with the cache that it finds, there are a few points that you need to know. There are a few times when the system is going to find the matching image, and then there are some times when it won't be able to find this image. Some of the rules when it comes to whether or not the cache is found include:

The system is going to try its best to locate the parent image if it is in existence. Then the instructions that come from that found image is going to be compared to the instructions of the child image from your main image. This is done in order to figure out if they have the same instructions or not. If this is not the case, the cache is going to be invalidated.

In many instances, you are able to compare the instructions that you see in the Dockerfile with one of the child images you are working with to see whether they are seen as sufficient enough. But there may be some times when the instructions will need a closer look at the images and more of an explanation before you can proceed.

When you are trying to add or copy the instructions, the contents of the file in the image are also going to be looked over, and then a checksum is going to be done on each and every file. The one that you accessed or modified last is going to be kept out of this checksum. During the lookup of the cache, the checksum will compare itself against the other checksums in the images that are present. This helps it to see if there is a match.

After the cache has been looked for and checked, and it is seen as invalidated, every Dockerfile command will need to start creating a new image causing the cache to not be used at all.

Instructions for the Dockerfile

Now, there are a lot of instructions that you are able to work with when you are on the Dockerfile program. You need to make sure that you are able to write out the instructions as well as possible. Some of the different things that you need to know about creating and writing out these instructions includes:

From

Any time that it is possible, you want to make sure that you are using the most recent of the official repositories for the basis of any new image that you would like to create. Debian image is often the best to work with because it is controlled tighter than some of the other options, and it will stay minimal, even during a full distribution. This helps you to get more out of the system without it being slow or dragging the system down.

Label

Labels are easy to add to the image and you would do this in order to organize the images of your project, to help with any automation processes that you have, and for many other reasons. Every label that you add is going to add a new line and it will begin with the label command. It should also have a minimum of one, but can certainly have more key value pairs.

Run

If you want to actually be able to use these images and get them to work well for you, you must make sure that the Dockerfile is readable. But how do we make sure that the file or the image is readable in this system? To do this, you must make sure that the run statements, the ones that are complex and long are put on more than one line, and that you take the time to separate them out by backslashes. This makes it easier for the system to read through the information that you have and can ensure that you will actually get the right commands read through by the Docker.

Apt-get

Now, when we are talking about run, the most common way to do this is going to be the application for the apt-get function. This is a function that will be responsible for installing packages and there will be a few gatchas that are observed in it. These are there to make sure that all parts of this process work the way they should.

If it is possible, you need to avoid using the run apt-get upgrade and the dist. –upgrade should be limited. The reason for this is because many of your parent images won't provide you with the right privileges. Should a package have a parent image and it is out of date, then you need to contact the maintainers to get this fixed. If you know that there is some kind of package present, and it needs to be updated, then you are able to use the apt–get install -y foo to make sure that everything updates on its own.

As a developer in Docker, you should try to combine the command for run apt-get update along with the apt-get install. When you use these with the run statements, you will sometimes run into trouble with a caching issue because you will get the install to fail in this process.

After you have had time to build an image, you will be able to look through all of the layers that are in a cache in the Docker system. You can even go through and modify these layers later on with a bit of extra packaging. To make some modifications, you will need to work with the apt-get install function to make it work, to get the right updates, and ensures that you don’t cause issues.

Remember that Docker is going to take note that sometimes, the instructions for modifications that you are using will be identical, and this means that you should reuse the cache that was processed in previous steps. When this happens, it will mean that this update won’t go through and get executed because the build is relying on the cached version instead.

Since this is an update that won’t run in this situation, it is possible that the build you are working on is outdated. When you decide to use the run apt-get update && apt-get install – y will make sure that the Dockerfile is installed and that the package you are working with is as up to date as possible. This is a nice code to work with because it is simple and ensures that you won’t need any additional code in the process. This is a process that is known as cache busting and you can easily get this to happen when you talk to the program and let it know which version you are in.

Using pipes

There are going to be some of the run commands out there that you may use that will also depend on the pipeline and its output of a single command into another by working with the pipe character. Docker will automatically choose to execute these particular commands by working with the /bin/sh -c interpreter. This is a good interpreter to work with because it can evaluate the exit code while looking over the

end of your code that was run in the pipe, checking out how successful it was at that time.

If you want to make it so that the command fails because of an error that is located anywhere along the pipe, then you are going to set the -o pipefail && command so that the error shows up and the build will not be able to succeed. One thing to note here is that not every shell you work with will support this command. There are some cases, like in dash, where the default shell on Debian based imaging is going to change the code. You could use the exec form of the run code in order to pick out a shell that will support the pipe fail option if you still want to work with it.

Cmd

The next thing that we need to look at is the Cmd instruction. This is something that you are going to use when you would like to run the software that is already found inside the image. You can also use it if there are arguments in the image and you would like to have those execute as well. Only work with the Cmd when you use it like this Cmd [“executable”, “ param1”, “param2”, ...].

Expose

Expose is going to indicate which of the ports are inside a container and it will take some time to listen to see if there are any connections present. At this same time, you may want to work with the traditional ports for the application. Let’s take Apache and an image that is found on it. This image can expose 80 and a different image is going to work with an expose of 28391.

When you use the expose feature, you and the other users who are on the program must go through and execute the run command in Docker, like we have talked about before,

with a flag attached to ensure that the ports are mapped in the proper manner. With each container that is linked, Docker will be able to provide you with an environmental variable on that path, which can lead the container back where it originally came from.

Env

And finally, we are going to take a look at the Env feature. To ensure that any newer software you add to the mix is easier to run, you can use the env function to update the environmental variable for the software that your container was able to install. An example of this is the following:

Env path/usr/local/nginx/bin:

The $path part of it is there to make sure that your Cmd, which is the [“nginx”] is going to work in the code the way that it should be. This form of instruction is going to be useful when you make sure that all of the variables are tied to the services that you would like them and in the right container.

And to finish off, env is also useful because it has the option of being used to help set the version numbers that you need to use the most often. This helps the version bumps to be easier to maintain overall. This is going to be similar to having these constant variables in the program.

Chapter 15: Is the Docker Cloud Important and More About Setting Up Your Docker Cloud Account

_____

We talked briefly about and mentioned it a few times, the Docker Cloud. This is an important part of working with the Docker program. It allows you to store the information that you make with the images and the containers that you work on and it allows various team members to get onto the system and work as well, all in one place. Being able to properly work with the Docker Cloud can be so important when it comes to making sure that you get the most out of this system. This chapter is going to take a look at some of the things that you need to know about the Docker Cloud and some of the ways that you can get the most out of the Cloud and the Docker system.

Cloud settings and the IDs

The cloud that you use with Docker is going to take a look at the Docker ID that you set up in order to link the cloud account and the account you have for the hub together. If you have an account with the Docker system already, you can simply use the login information for that account in order to sign in with the cloud and use those features.

But if you don’t have one of these Docker IDs, this isn’t a big deal. You just need to sign up for one by going to the

website for the Docker cloud, or you can log in using the command prompt on the system. Make sure that the name and ID that you pick out is one that you like because it becomes the namespace for the whole account, and you won’t be able to change it without starting a new account and losing everything.

Managing the cloud services

Your hosts can easily be linked over to the Docker clouds as well as to the nodes that are offered in the Docker Cloud. So if you are already a member with the Amazon cloud or the Microsoft cloud, you are able to link these back to Docker through the cloud services. There are even some other services, such as Bitbucket and GitHub that you can link back to the cloud.

Take some time to look through the different services that you are able to add in with your Docker Cloud. The more that you can link back, the easier your work can be. In fact, this can save you a lot of time and resources to get everything moved over by linking the right cloud services back to the Docker Cloud.

Email addresses

There are several different email addresses that you are able to add to Docker, but you need to have a primary one so that notifications can be sent to that one instead of you having to go through all of the emails for an email from Docker. If you are on the cloud and you want to be able to add in multiple email addresses, you can use the steps below to help you get these emails in place:

Choose the user icon. You can see this at the top of the screen before you move on to the account settings.

Go to the section that is entitled emails. This helps you to add in any of the emails that you need.

Select the plus sign so that you can get an email verifying your email before you move on.

New emails won't be added to the system until you have been able to confirm them. There will be a link that can be live for a short time period before you will need to have a new email sent for the verification process, so try to do this right away. Should you want to add in a few email addresses that you want to be verified by the program, you can select the primary email address to make sure that these emails are going to the right location.

Notifications

These accounts are going to be configured in a way so that there are emails sent out any time that a new event occurs in the cloud. Slack also has the option of being linked to this

cloud so if you do this, then the notifications will come to your inbox from there.

Paid accounts

When you are looking through the cloud of Docker, you will notice that there are two options. You can work with a paid account or a free account. If you choose the free account, you will notice there are some restrictions that come with it. For example, those on a free account will be able to just work on a single repository. If you are just trying out the system, you may want to try out the free choice. But if you need more and really want to see what comes with the Docker program, then you may want to go with the paid account. This will allow you to go through and choose your settings and then go with a plan that works the best for your needs with Docker.

Teams and organization for the cloud

Organizations that go on the Docker cloud will be able to share a lot of things including the repositories and the infrastructure that other applications are going to use. Members from that organization can also post in the team that you have placed them in, and they will be able to see what kind of status they have inside that organization on the team.

If a member starts an organization and is the first one on the program, they will be able to modify anything that occurs in that organization. This will include the team as well as the membership status of those who are in the team. If you are not in supposed to be in the organization, you will not have access to see anything that is inside of it, including the users and which team they belong to, for security reasons.

Creating your organization

Organizations that occur on the Docker cloud are going to be composed of teams and then each of these teams will have however many users deemed necessary to make the team work the way that the owner wants. This could be anywhere from two to more users in a team. However, the users can’t be added straight into the organization. This is because many of the organizations are going to have applications, infrastructure, repositories, and other things that will be associated with them and that members of the team will be able to access. This gives a bit more control over who gets to do what.

Before you are able to use one of these organizations, you must make sure that you create the organization in the proper manner. The steps that you need to follow to make this happen include:

Log into the cloud with the username and password. You should already have these from setting up your Docker account.

Create an organization from the icon that you should be able to find near the top, on the right side of that page.

Name the organization. You want to make sure that the name fits with the tasks, or with your business, or makes sense in some other manner. You will have this name placed in the dialog box that is on your screen.

Insert the right billing information. This is required whether you are going to use the paid version or not and makes it easier if you are going to use some of the services that you can pay for.

Save the organization. The cloud will take some time to switch you over to the organization view so that you can look at everything. You can also go back to the private account if you want to use it outside of the organization that was created.

When the organization is done in the cloud, you can go through and add that person's account to the owner's team. This is done automatically by the program that you are using. Your team has to include one or more other people, ones that aren't you. You can always go through and add more or remove these later, but there should always be a minimum of one other person on the account.

Converting the user over to an organization

There may be times when you already have an individual account through the Docker cloud and you will want to convert it into an organization. The account is no longer going to be set up in a way that you can log into it with an email address or other things that link back to this account to make sure that you are able to keep your security. You will also see that any collaborators on that account are going to be removed. The builds that are already automated are going to be moved over and then, once you have time

to convert your account, you won’t be able to go backward and turn it back into a personal account.

Now, if you want to convert your user account over to a new organization, you must make sure that you have an ID that is valid for the account. This is because the user name is going to show up as the first member of that team. From there, any build that is on the account will migrate over to this user, who can then convert them to the settings that the organization needs, which allows other users you invite to actually work on and use them.

If you would like to change your personal user account into an organizational account through the Cloud, some of the steps that you will need to work with include the following:

Log into the Docker Cloud using the account information that you would like to convert.

Head over to your settings on that account.

You can look for the option about converting over to an organization and then select the option.

Make sure that you read through any warnings that do happen to show up. This will help you know what is going on and may remind you to save some things or do other important steps to protect your information.

After reading the warnings, you can insert the Docker ID for the first member of the group to get that set up.

Make sure to save the changes before you continue.

After you do this, you will find the system will refresh and then you can go through and log into this using the same ID that you choose for the organization earlier on. You can then go in and do all of the configurations that are needed for the organization.

Giving team members certain access

When you are working with the Docker Cloud, you will find that you can choose the level of permission that each team member has. If you want one person to be able to do more than the others, then you can set this up. If you want one group to be able to have certain privileges, you can do that as well. There are actually a few different access levels that you can give to your teams and these include:

Admin: these are the individuals who will be able to manage the resources and the permissions on the organization. They will be able to look at and see all of the stuff that occurs in the team. The Admin position is usually going to be reserved for the individual who owns the team, but this permission can be changed to include other people if you choose to do this option.

No access: This is one where the user isn’t able to do anything and all of the resources are going to be invisible. This is the strictest form of access and it is not one that you will use all that often unless you want to slow down the progress of the team.

Read and write: This is where you allow the users to view and modify the resources that are found in the team.

Read only: Users who have this kind of permission will be able to read through the resources that are in the system and on the cloud, but they won’t be able to do anything with this.

How to change the permissions for an individual repository

There are a lot of different ways that you are able to change up the permissions on the different parts of your program and this includes with an individual repository. Permissions can be given out for just one repository for the team to have some access to it using the permissions page, rather than having to take the time to go through the settings of each member and then try to change them. This can be done in the event that the repository is going to be made at a time after this team has started and the access will need to be given to more than one team at a time.

The repository for an organization can be set at private as well. When you set it up as this way, you will need to personally go through and grant the right access to any team member that you would like to have that permission. If you want to allow all of the team members to have access to that one repository, then you must make it public. But then they will be in the default mode of only having read-only access to that program.

Anyone who is part of your organization that has administrative permissions will be able to go into the account and change up the permissions for anyone who is on the team, at any time that they want. You will have to either limit who is considered an admin or realize who can do what.

If you would like to grant a certain team some access to get into one of your repositories, there are a few steps that they need to follow. These steps will include the following:

Head over to the repository that is in your organization and that you want to work with.

Choose the tab that lists it for permissions.

Select the team that you want to change the permissions for or that you would like to see the repository in question.

Choose what access level you would like to give to the team when they come in to that area.

Click on what appears to be a plus sign by those who need to be able to access the resources freely. Once you click on that sign, the changes are going to be saved automatically.

Make sure that you go through these steps and redo them for all of the teams that you want to have this kind of access.

When we are talking about making some edits to the permissions you have, you will need to head over to a new settings menu. You will see this pop up when you get to the permissions page. Deleting the permissions that someone automatically has, or that you gave them, is fairly easy. All that has to happen here is to click on the trash can that is located next to the name of the team. This action is going to work at getting rid of the access the entire team has. This means that any permission they were given earlier is going to be gone.

One thing to note here is that when we are looking at public repositories, this is going to grant access that is read-only automatically. However, you will find that it can change over to private as a way to stop those who don't have the right permissions in place from seeing what is inside of it.

Modify the team

There are times when you will want to make some changes to the team. A team that is already up and running can be changed. You simply just need to log into the cloud and then switch yourself over to the admin console of the organization. Going to the team section, you will then be able to select the team that you want to make some changes to.

You are able to manage all of the different memberships of your team on the first page. You just need to head over to the team that you want to change and you can see the management tools right on that page. To make any

changes, like making modifications to the description or the name of the team, you can just go to the settings. But if you want to change up the permissions that the team has for a project or within the organization, then you must head over to the permission page to do this.

How to manage the resources for the organization

The final topic that we are going to take a look at here for the cloud is how to manage the different resources that you have for the organization. Each organization that gets set up will have some of its own resources to pull from. These resources may include services, containers, and nodes to name a few. The organizations are just larger user accounts that have more people, and you can use the resources that come from each one.

Owners are still going to be the team members, but they are able to create all of the resources required, allowing the organization as a whole to utilize all of these resources. The resources will be crafted as soon as you can log into the page that is tied to the organization, and then you can manage what the team is able to see, and what will remain private.

This brings up the question, how to link the services into the cloud and the account for the organization to use them? Some of the steps that you need to make this happen include:

Log into your cloud. Make sure that you are doing this as the owner of the organization, rather than as one of the team members.

Locate the account that you have associated with this organization.

Find the settings. You should be able to see these on the cloud and then link the source codes for the organization to the cloud.

These are the same steps that you need to follow in order to give certain special permissions for a single person. You can determine what each person in your organization is able to see and what they are able to do with that organization to keep things organized.

As you will find, working with the Docker Cloud is one of the best things that you can work with in order to see some results. The Cloud is going to help you to hold all of your information in one place and make sure that you are able to share that information with others who are on the system as well. Make sure to take a look at some of these steps to ensure that you are able to get the most out of using the Docker system.

Chapter 16: Tools to Consider Using with Docker to Make Things Easier

_____

When it comes to being able to use Docker in an effective manner, you will only be as good as some of the tools that you choose to put towards the task at hand. With this in mind, consider some of the tools below to make sure that you are able to fully use Docker in the way that it is meant and that you can take your programming game with the Docker system to the next level.

Container migration tool: This is a tool that can be used in the command line and used to both runC and Docker. You can use it in order to migrate some of your live containers between different hosts and it works through the use of validations that must occur before the migration. This helps you to discover the additional target hosts on its own. If you would like to move containers around and many of them on a regular basis, then this is a tool that you should use.

Docker label inspector: This is a tool that you can use when you want to add relevant metadata to containers that you then plan to push over to the Docker hub. This is a great tool that you can use in order to apply predetermined labels to swarm services, networks, volumes, swarm nodes, local

daemons, images, and containers. This metadata can then be created in the domain of existing container technology as well as check the labels compared to the official schema. It is also able to validate the tags against any of the JSON schema that is provided.

Dvol: The dvol tool is going to make it easier for you to control the specific version of the database that you are trying to work on in Docker. This is a tool that can help you to branch, commit, and reset the different databases so that it is easier for you to save the state that you want and then come back to it later on. This tool can also integrate with the Docker Compose feature so it is easier to spin the microservice environments that are easy to reproduce later on if needed.

Libnetwork: This is a tool that is able to combine together the engine of Docker and the libcontainer networking code. This provides you with a useful library for networking various containers and it is really multiplatform. The goal of using this tool is to provide a network model for the containers that are strong and that provides access to a consistent interface for programming. It also has a lot of plugin and driver options that you can choose to work with.

Wagl: This is a tool that will allow any of the DNS servers that are running the microservices if they were containers that are a part of a swarm on Docker to locate and also to communicate with each other. This is designed to be minimalist and works similar to a drop-in container that you

are able to add into any cluster that needs the DNS based service and discovery.

Prometheus: This is a tool that you need to have when you want to remove some of the stigmas that is there about the black box nature of images that you were able to produce in a type of vacuum and then later pushed over to the Docker hub. This is basically an open source framework that Soundcloud originally made. And its original purpose is to monitor and provide some analytics for the containers in Docker that don’t have this information already built in.

Docker Compose: This is a deployment tool that is useful when it comes to testing, developing, and staging the multi-container applications inside of the Docker system. This means that this tool is able to manage the entire part of the application through all of its lifecycles. Setting up this program is easy because all that you need is a basic configuration file to get it started. Once the configuration is done, you can then use one command to get this application up and running.

Flocker: This is a volume orchestrator that is used with open source data for containers. Unlike some of the other similar products, Flocker is seen as more portable and it is not going to be tied down to one of the servers over another. This means that you are able to use it on any container that is in the cluster. It is also there to help you to migrate data between any host as you need and it is going to integrate nicely with Docker.

Drone: Drone is a platform that is going to promote a continuous amount of testing and it is able to integrate with several different platforms including Github, Amazon, Heroku, and Google AppEngine based on your needs. This tool can also help you to create some custom containers and will allow you to run these tests from your own computer or your own network. You can set up some custom workflows through this and then automatically build and deploy any of the containers that are needed when you work on coding.

Dockersh: This is a tool that was created by Yelp as a way to service and test the management of its infrastructure suite. It is going to work because it provides each of the containers that you create in Docker with a user that corresponds to it. The users are able to view just their own home directory and make any of the changes that are necessary. There will also be some limits as to what each user is able to see. The limitations that you can place with this will help to mitigate any potential concerns that may arise when you try to give your users some direct access to a shell that is already there.

These are just a few of the different tools that you can choose to work with when you are creating your own Docker system. There are many more out there and choosing one can often be difficult. You can look through these and more and decide which plugins and tools are going to be the most useful based on what you plan to use the Docker system for.

Conclusion

_____

Thank for making it through to the end of Docker Tutorial for Beginners. Let’s hope it was informative and able to provide you with all of the tools you need to achieve your goals whatever they may be.

The next step is to start implementing some of the different aspects of Docker that we have talked about in this guidebook. As someone who has heard about Docker and who knows some of the great benefits that come with using this system, the next step is to actually learn how to use the system and how to make it work for your own organization. This guidebook took some time to look through the different strategies and aspects of the Docker system that you need to learn in order to get the most out of it.

No matter why you choose to use the Docker system or your experience with programming and other networking systems, we hope that this guidebook has given you the tools that you need to see some results.

Finally, if you found this book useful in any way, a review is always appreciated!