**Docker安装**

**一.容器**

* 1. 什么是容器？

容器就是将软件打包成标准化单元，以用于开发、交付和部署。

* 1. 容器的特点
* 容器镜像是轻量的、可执行的独立软件包 ，包含软件运行所需的所有内容：代码、运行时环境、系统工具、系统库和设置。
* 容器化软件适用于基于Linux和Windows的应用，在任何环境中都能够始终如一地运行。
* 容器赋予了软件独立性　，使其免受外在环境差异（例如，开发和预演环境的差异）的影响，从而有助于减少团队间在相同基础设施上运行不同软件时的冲突。

1.3 Docker与VM虚拟机区别

* VM(VMware)在宿主机器、宿主机器操作系统的基础上创建虚拟层、虚拟化的操作系统、虚拟化的仓库，然后再安装应用；
* Docker守护进程可以直接与主操作系统进行通信，为各个Docker容器分配资源；它还可以将容器与主操作系统隔离，并将各个容器互相隔离。虚拟机启动需要数分钟，而Docker容器可以在数毫秒内启动。由于没有臃肿的从操作系统，Docker可以节省大量的磁盘空间以及其他系统资源。
* 两者有不同的使用场景。虚拟机更擅长于彻底隔离整个运行环境。例如，云服务提供商通常采用虚拟机技术隔离不同的用户。而Docker通常用于隔离不同的应用，例如前端，后端以及数据库。
* Container(Docker容器)，在宿主机器、宿主机器操作系统上创建Docker引擎，在引擎的基础上再安装应用。

Docker中创建一个系统的镜像文件，这样就能将系统集成到Docker中，运行的应用就都是该系统的应用。

可以认为是一个轻量级的虚拟机。

**二.** **Docker基本概念**

* 1. Docker基本概念？
* 镜像(Image)
* 容器(Container)
* 仓库(Repository)

**1.1.1** 镜像（Image）—— 一个特殊的文件系统

操作系统分为内核和用户空间。对于 Linux 而言，内核启动后，会挂载 root 文件系统为其提供用户空间支持。而Docker 镜像（Image），就相当于是一个 root 文件系统。

Docker 镜像是一个特殊的文件系统，除了提供容器运行时所需的程序、库、资源、配置等文件外，还包含了一些为运行时准备的一些配置参数（如匿名卷、环境变量、用户等）。 镜像不包含任何动态数据，其内容在构建之后也不会被改变。

**1.1.2** 容器（Container)——镜像运行时的实体

镜像（Image）和容器（Container）的关系，就像是面向对象程序设计中的 类 和 实例 一样，镜像是静态的定义，容器是镜像运行时的实体。容器可以被创建、启动、停止、删除、暂停等 。

容器的实质是进程，但与直接在宿主执行的进程不同，容器进程运行于属于自己的独立的 命名空间。前面讲过镜像使用的是分层存储，容器也是如此。

**1.1.3** 仓库（Repository）——集中存放镜像文件的地方

镜像构建完成后，可以很容易的在当前宿主上运行，但是， 如果需要在其它服务器上使用这个镜像，我们就需要一个集中的存储、分发镜像的服务，Docker Registry就是这样的服务。

一个 Docker Registry中可以包含多个仓库（Repository）；每个仓库可以包含多个标签（Tag）；每个标签对应一个镜像。所以说：镜像仓库是Docker用来集中存放镜像文件的地方类似于我们之前常用的代码仓库。

最常使用的 Registry 公开服务是官方的 Docker Hub ，这也是默认的 Registry，并拥有大量的高质量的官方镜像，网址为：https://hub.docker.com/ 。在国内访问Docker Hub 可能会比较慢国内也有一些云服务商提供类似于 Docker Hub 的公开服务。比如 时速云镜像库、网易云镜像服务、DaoCloud 镜像市场、阿里云镜像库等。

**三.Docker的安装**

Docker从1.13版本之后采用时间线的方式作为版本号，分为社区版CE和企业版EE。

社区版是免费提供给个人开发者和小型团体使用的，企业版会提供额外的收费服务，比如经过官方测试认证过的基础设施、容器、插件等。

1. **CentOS版本安装**

目前，CentOS 仅发行版本中的内核支持 Docker。

Docker 运行在 CentOS 7 上，要求系统为64位、系统内核版本为 3.10 以上。

Docker 运行在 CentOS-6.5 或更高的版本的 CentOS 上，要求系统为64位、系统内核版本为 2.6.32-431 或者更高版本。

**1.1.1** **查看系统版本**

Docker 要求 CentOS 系统的内核版本高于 3.10 ，查看本页面的前提条件来验证你的CentOS 版本是否支持 Docker 。

通过 uname -r 命令查看你当前的内核版本

|  |
| --- |
| [root@bogon ~]# uname -r |

**1.1.2** **将系统包更新到最新版本。**

|  |
| --- |
| [root@bogon ~]# yum update |

**1.1.3** **安装一些必要的系统工具。**

|  |
| --- |
| [root@bogon ~]# sudo yum install -y yum-utils device-mapper-persistent-data lvm2 |

**1.1.4** **配置docker仓库**

|  |
| --- |
| [root@bogon ~]# sudo yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo |
| 报以下错误:  Loaded plugins: fastestmirror  adding repo from: https://download.docker.com/linux/centos/docker-ce.repo  grabbing file https://download.docker.com/linux/centos/docker-ce.repo to /etc/yum.repos.d/docker-ce.repo  Could not fetch/save url https://download.docker.com/linux/centos/docker-ce.repo to file /etc/yum.repos.d/docker-ce.repo: [Errno 14] curl#60 - "Peer's Certificate has expired." |

**这是由于国内访问不到docker官方镜像的缘故 可以通过aliyun的源来完成：**

|  |
| --- |
| [root@bogon ~]# sudo yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo; |
| 出现以下内容则表示docker仓库配置成功：  Loaded plugins: fastestmirror  adding repo from: http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo  grabbing file http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo to /etc/yum.repos.d/docker-ce.repo  repo saved to /etc/yum.repos.d/docker-ce.repo |

**1.1.5** **安装docker**

|  |
| --- |
| [root@bogon ~]# sudo yum install docker-ce |
| 出现以下异常：  Loaded plugins: fastestmirror  base  https://download-stage.docker.com/linux/centos/7/x86\_64/stable/repodata/repomd.xml: [Errno 14] curl#35 - "TCP connection reset by peer"  Trying other mirror.  One of the configured repositories failed (Docker CE Stable - x86\_64),  and yum doesn't have enough cached data to continue. At this point the only  safe thing yum can do is fail. There are a few ways to work "fix" this:  1. Contact the upstream for the repository and get them to fix the problem.  2. Reconfigure the baseurl/etc. for the repository, to point to a working  upstream. This is most often useful if you are using a newer  distribution release than is supported by the repository (and the  packages for the previous distribution release still work).  3. Run the command with the repository temporarily disabled  yum --disablerepo=docker-ce-stable ...  4. Disable the repository permanently, so yum won't use it by default. Yum  will then just ignore the repository until you permanently enable it  again or use --enablerepo for temporary usage:  yum-config-manager --disable docker-ce-stable  or  subscription-manager repos --disable=docker-ce-stable  5. Configure the failing repository to be skipped, if it is unavailable.  Note that yum will try to contact the repo. when it runs most commands,  so will have to try and fail each time (and thus. yum will be be much  slower). If it is a very temporary problem though, this is often a nice  compromise:  yum-config-manager --save --setopt=docker-ce-stable.skip\_if\_unavailable=true  failure: repodata/repomd.xml from docker-ce-stable: [Errno 256] No more mirrors to try.  https://download-stage.docker.com/linux/centos/7/x86\_64/stable/repodata/repomd.xml: [Errno 14] curl#35 - "TCP connection reset by peer" |

**分析原因为：阿里的镜像库文件也指向docker官方库，所以需要修改库文件**

|  |
| --- |
| [root@bogon ~]# sudo vim /etc/yum.repos.d/docker-ce.repo |
| 通过命令把https://download-stage.docker.com替换为http://mirrors.aliyun.com/docker-ce  命令如下：:%s#https://download-stage.docker.com#http://mirrors.aliyun.com/docker-ce#g  再执行docker的安装命令即可安装成功。 |

**出现如下提示、则安装成功**

|  |
| --- |
| Installed:  docker-ce.x86\_64 0:18.03.0.ce-1.el7.centos  Dependency Installed:  audit-libs-python.x86\_64 0:2.7.6-3.el7 checkpolicy.x86\_64 0:2.5-4.el7 container-selinux.noarch 2:2.42-1.gitad8f0f7.el7 libcgroup.x86\_64 0  libtool-ltdl.x86\_64 0:2.4.2-22.el7\_3 pigz.x86\_64 0:2.3.3-1.el7.centos policycoreutils-python.x86\_64 0:2.5-17.1.el7 python-IPy.noarch  Complete! |

**1.1.6 验证docker安装成功**

**启动docker:**

|  |
| --- |
| sudo systemctl start docker |

**验证docker:**

|  |
| --- |
| sudo docker run hello-world |
| 出现以下异常:  Unable to find image 'hello-world:latest' locally  latest: Pulling from library/hello-world  9bb5a5d4561a: Pulling fs layer  docker: error pulling image configuration: Get https://dseasb33srnrn.cloudfront.net/registry-v2/docker/registry/v2/blobs/sha256/e3/e38bc07ac18e  See 'docker run --help'. |

**此错误也是网络问题：出现这个问题原因为国内网络问题，无法连接到 docker hub。**

**国内已经有  daocloud，docker指定该源即可。**

|  |
| --- |
| vi /usr/lib/systemd/system/docker.service  [Unit]  Description=Docker Application Container Engine  Documentation=https://docs.docker.com  After=network-online.target firewalld.service  Wants=network-online.target  [Service]  Type=notify  # the default is not to use systemd for cgroups because the delegate issues still  # exists and systemd currently does not support the cgroup feature set required  # for containers run by docker  ExecStart=/usr/bin/dockerd **--registry-mirror=http://f2d6cb40.m.daocloud.io**  ExecReload=/bin/kill -s HUP $MAINPID  # Having non-zero Limit\*s causes performance problems due to accounting overhead  # in the kernel. We recommend using cgroups to do container-local accounting.  LimitNOFILE=infinity  LimitNPROC=infinity  LimitCORE=infinity  # Uncomment TasksMax if your systemd version supports it.  # Only systemd 226 and above support this version.  #TasksMax=infinity  TimeoutStartSec=0  # set delegate yes so that systemd does not reset the cgroups of docker containers  Delegate=yes  # kill only the docker process, not all processes in the cgroup  KillMode=process  # restart the docker process if it exits prematurely  Restart=on-failure  StartLimitBurst=3  StartLimitInterval=60s  [Install]  WantedBy=multi-user.target |
|  |

**重启docker服务：**

|  |
| --- |
| $ sudo systemctl daemon-reload  $ sudo systemctl restart docker |

**再次验证docker:**

|  |
| --- |
| sudo docker run hello-world |

**出现一下内容表示验证成功：**

|  |
| --- |
| Unable to find image 'hello-world:latest' locally  latest: Pulling from library/hello-world  9bb5a5d4561a: Pull complete  Digest: sha256:f5233545e43561214ca4891fd1157e1c3c563316ed8e237750d59bde73361e77  Status: Downloaded newer image for hello-world:latest  Hello from Docker!  This message shows that your installation appears to be working correctly.  To generate this message, Docker took the following steps:  1. The Docker client contacted the Docker daemon.  2. The Docker daemon pulled the "hello-world" image from the Docker Hub.  (amd64)  3. The Docker daemon created a new container from that image which runs the  executable that produces the output you are currently reading.  4. The Docker daemon streamed that output to the Docker client, which sent it  to your terminal.  To try something more ambitious, you can run an Ubuntu container with:  $ docker run -it ubuntu bash  Share images, automate workflows, and more with a free Docker ID:  https://hub.docker.com/  For more examples and ideas, visit:  https://docs.docker.com/engine/userguide/ |

**Docker卸载:**

|  |
| --- |
| yum remove docker \  docker-client \  docker-client-latest \  docker-common \  docker-latest \  docker-latest-logrotate \  docker-logrotate \  docker-selinux \  docker-engine-selinux \  docker-engine  rm -rf /etc/systemd/system/docker.service.d  rm -rf /var/lib/docker  rm -rf /var/run/docker |