链滴

podman 初体验

作者：yuanhenglizhen

原文链接：https://ld246.com/article/1575638404861

来源网站：链滴

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

1. 概述

podman官网

https://podman.io/getting-started/

写这个博客的原因是在地铁上，看到一篇博客《Docker 大势已去，Podman 万岁》，就去了解了下pdman，Redhat的亲儿子。毕竟拥抱开源，拥抱k8s，pod的概念也有。

很多人可能遇到过开机重启时，由于Docker守护程序在占用多核CPU使用100%C使用的情况，导致有容器都无法启动，服务都不能用的情况。过利用Docker重构WP博客的新架构。由于VPS机器不是稳定，时常会重启，重启时候就会遇到这个事情，VPS负载很高，容器都没有起来，网站就无法访问。这时候只能杀掉所有容器并重启守护进程，才能恢复。经过了解该问题是由于Docker守护进程引，而且Docker守护进程是以root特权权限启动的，是一个安全问题，那么有什么方法解决呢?

为什么Docker需要一个守护进程呢?

Podman，Skopeo和Buildah

这三个工具都是符合OCI计划下的工具（github/containers）。主要是由RedHat推动的，他们配合以完成Docker所有的功能，而且不需要守护程序或访问有root权限的组，更加安全可靠，是下一代器容器工具

2. 准备环境

这边我购买了一台阿里云2核8G的ECS，抢占式付费模式。抢占式付费模式，比按时付费便宜，这边置下最高竞价0.5元每小时，可以保障机器的所有权。选择aliyun linux，也就是o的一个系统，系统身做了一些优化之类的。

下面这张图就比较有意思了，现在很少可以买到经典网络的ECS了，除非之前有经典网络的ECS，否默认都是专有网络。阿里云大概从2016年开始，就是按区域做网络迁移了，增强了网络隔离性。还提设备更换，性能更好，这就不知道了。恰巧前几天，碰上了网络模式迁移这件事。有几点需要注意下就是

1.RDS网络模式改成专有网络，首先需要打开白名单高安全模式（经典网络和VPC隔离），如果之前

应用链接的时候，最好迁移的时候不保留经典网络，这样链接地址不会变，不然需要修改RDS链接地。保留的会就需要打开混淆模式，经典网络和VPC打通访问，当然这个因应用而异，也许有更无感的移方式。

2.ECS迁移，可以保留原内网地址，但是需要你建的VPC网络，包含之前内网的网段，待迁移完成之，自行手动修改

等到所有的操作都点完之后，我们服务器环境就准备好啦

3. 安装podman

3.1 更新操作系统

yum update -y

体验了一把速度还是很快的，一会会就更新完毕了

3.2安装podman

sudo yum -y install podman

已安装:
  podman.x86_64 0:1.4.4-2.1.al7

原文链接：podman 初体验

```
作为依赖被安装:
  PyYAML.x86_64 0:3.10-11.1.al7                    atomic-registries.x86_64 1:1.22.1-26.gitb5070
9.1.al7            audit-libs-python.x86_64 0:2.8.5-4.1.al7
  checkpolicy.x86_64 0:2.5-8.1.al7                 container-selinux.noarch 2:2.107-1.1.al7
          containernetworking-plugins.x86_64 0:0.8.1-1.1.al7
  containers-common.x86_64 1:0.1.37-1.1.al7          criu.x86_64 0:3.12-2.1.al7
          libcgroup.x86_64 0:0.41-21.1.al7
  libnet.x86_64 0:1.1.6-7.1.al7                    libsemanage-python.x86_64 0:2.5-14.1.al7
          libyaml.x86_64 0:0.1.4-11.1.al7
  policycoreutils-python.x86_64 0:2.5-33.1.al7       protobuf-c.x86_64 0:1.0.2-3.4.al7
            python-IPy.noarch 0:0.75-6.1.al7
  python-backports.x86_64 0:1.0-8.1.al7              python-backports-ssl_match_hostname.noa
ch 0:3.5.0.1-1.1.al7      python-ipaddress.noarch 0:1.0.16-2.4.al7
  python-pytoml.noarch 0:0.1.14-1.git7dea353.1.al7      python-setuptools.noarch 0:0.9.8-7.1.a
7                    runc.x86_64 0:1.0.0-64.rc8.1.al7
  setools-libs.x86_64 0:3.3.8-4.1.al7

完毕!
```

4. 探索podman命令

4.1 podman -h

可以大概看出了，和docker的命令几乎是一致的

```
manage pods and images

Usage:
  podman [flags]
  podman [command]

Available Commands:
  attach      Attach to a running container
  build       Build an image using instructions from Dockerfiles
  commit      Create new image based on the changed container
  container   Manage Containers
  cp          Copy files/folders between a container and the local filesystem
  create      Create but do not start a container
  diff        Inspect changes on container's file systems
  events      Show podman events
  exec        Run a process in a running container
  export      Export container's filesystem contents as a tar archive
  generate    Generated structured data
  healthcheck Manage Healthcheck
  help        Help about any command
  history     Show history of a specified image
  image       Manage images
  images      List images in local storage
  import      Import a tarball to create a filesystem image
  info        Display podman system information
  init        Initialize one or more containers
  inspect     Display the configuration of a container or image
  kill        Kill one or more running containers with a specific signal
```

```
  load        Load an image from container archive
  login       Login to a container registry
  logout      Logout of a container registry
  logs        Fetch the logs of a container
  mount       Mount a working container's root filesystem
  pause       Pause all the processes in one or more containers
  play        Play a pod
  pod         Manage pods
  port        List port mappings or a specific mapping for the container
  ps          List containers
  pull        Pull an image from a registry
  push        Push an image to a specified destination
  restart     Restart one or more containers
  rm          Remove one or more containers
  rmi         Removes one or more images from local storage
  run         Run a command in a new container
  save        Save image to an archive
  search      Search registry for image
  start       Start one or more containers
  stats       Display a live stream of container resource usage statistics
  stop        Stop one or more containers
  system      Manage podman
  tag         Add an additional name to a local image
  top         Display the running processes of a container
  umount      Unmounts working container's root filesystem
  unpause     Unpause the processes in one or more containers
  unshare     Run a command in a modified user namespace
  version     Display the Podman Version Information
  volume      Manage volumes
  wait        Block on one or more containers

Flags:
      --cgroup-manager string       Cgroup manager to use (cgroupfs or systemd) (default "syst
md")
      --cni-config-dir string       Path of the configuration directory for CNI networks
      --config string               Path of a libpod config file detailing container server configuratio
 options
      --conmon string               Path of the conmon binary
      --cpu-profile string          Path for the cpu profiling results
      --default-mounts-file string  Path to default mounts file
      --help                        Help for podman
      --hooks-dir strings           Set the OCI hooks directory path (may be set multiple times)
      --log-level string            Log messages above specified level: debug, info, warn, error, fatal
or panic (default "error")
      --namespace string            Set the libpod namespace, used to create separate views of th
 containers and pods on the system
      --network-cmd-path string     Path to the command for configuring the network
      --root string                 Path to the root directory in which data, including images, is store


      --runroot string              Path to the 'run directory' where all state information is stored
      --runtime string              Path to the OCI-compatible binary used to run containers, defaul
 is /usr/bin/runc
      --storage-driver string       Select which storage driver is used to manage storage of imag
s and containers (default is overlay)
```

```
      --storage-opt stringArray      Used to pass an option to the storage driver
      --syslog                       Output logging information to syslog as well as the console
      --tmpdir string                Path to the tmp directory
      --trace                        Enable opentracing output
      --version                      Version for podman

Use "podman [command] --help" for more information about a command.
```

4.2 常用命令

4.2.1 查看运行的容器

```
[root@podman ~]# podman ps
CONTAINER ID  IMAGE  COMMAND  CREATED  STATUS  PORTS  NAMES
```

4.2.2 查看podman信息

```
[root@podman ~]# podman info
host:
  BuildahVersion: 1.9.0
  Conmon:
    package: podman-1.4.4-2.1.al7.x86_64
    path: /usr/libexec/podman/conmon
    version: 'conmon version 0.3.0, commit: unknown'
    ........
```

4.2.3 运行一个简单的容器

```
podman run -dt -p 8080:8080/tcp -e HTTPD_VAR_RUN=/var/run/httpd -e HTTPD_MAIN_CON
_D_PATH=/etc/httpd/conf.d                    -e HTTPD_MAIN_CONF_PATH=/etc/httpd/conf
      -e HTTPD_CONTAINER_SCRIPTS_PATH=/usr/share/container-scripts/httpd/      httpd
```

这边我们发现网速有点慢啊，找个国内源试下

```
mv /etc/containers/registries.conf  /etc/containers/registries.conf.bak
touch /etc/containers/registries.conf
cat > /etc/containers/registries.conf <<EOF
unqualified-search-registries = ["docker.io"]
[[registry]]
prefix = "docker.io"
location = "******.mirror.aliyuncs.com"
EOF
```

再次运行启动容器命令,就很快了

```
[root@podman containers]# podman run -dt -p 8080:8080/tcp -e HTTPD_VAR_RUN=/var/run
httpd -e HTTPD_MAIN_CONF_D_PATH=/etc/httpd/conf.d                    -e HTTPD_MAIN_CONF_
ATH=/etc/httpd/conf                  -e HTTPD_CONTAINER_SCRIPTS_PATH=/usr/share/container
scripts/httpd/      httpd
Trying to pull docker.io/library/httpd...
Getting image source signatures
Copying blob 000eee12ec04 done
```

```
Copying blob 32b8712d1f38 done
Copying blob 51c60bde4d46 done
Copying blob f1ca037d6393 done
Copying blob c4bd3401259f done
Copying config 2ae34abc2e done
Writing manifest to image destination
Storing signatures
```

4.2.4 查看镜像和容器

```
[root@podman containers]# podman ps
CONTAINER ID  IMAGE                           COMMAND         CREATED        STATUS            POR
S                NAMES
845ff52751b4  docker.io/library/httpd:latest  httpd-foreground  24 seconds ago  Up 23 secon
s ago  0.0.0.0:8080->8080/tcp  eager_ganguly
[root@podman containers]# podman images
REPOSITORY              TAG      IMAGE ID       CREATED      SIZE
docker.io/library/httpd   latest   2ae34abc2ed0   7 days ago   170 MB
```

我们访问下页面试下

```
[root@podman containers]# curl -iL 127.0.0.1:8080
curl: (7) Failed connect to 127.0.0.1:8080; 拒绝连接
[root@podman containers]# netstat -ntpl
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:5355            0.0.0.0:*               LISTEN      6432/systemd-resolv
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN      31319/conmon
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      6361/sshd
```

What?访问被拒绝，那就查看容器日志吧

```
[root@podman containers]# podman logs -f 845ff52751b4
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using
0.88.0.3. Set the 'ServerName' directive globally to suppress this message
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using
0.88.0.3. Set the 'ServerName' directive globally to suppress this message
[Fri Dec 06 12:36:57.794386 2019] [mpm_event:notice] [pid 1:tid 140331807290496] AH00489:
Apache/2.4.41 (Unix) configured -- resuming normal operations
[Fri Dec 06 12:36:57.794512 2019] [core:notice] [pid 1:tid 140331807290496] AH00094: Comm
nd line: 'httpd -D FOREGROUND'
```

进入容器查看,可能是文件的问题sweat_smile

```
[root@podman containers]# podman exec -it 845ff52751b4  /bin/bash
root@845ff52751b4:/usr/local/apache2# ll
```

那就简单暴力点吧

```
podman run -d -p 80:80 httpd
[root@podman containers]# curl 127.0.0.1
<html><body><h1>It works!</h1></body></html>
[root@podman containers]#
```

这边可以看到容器启动并访问成功了

```
[root@podman containers]# podman inspect 84 | grep IPAddress\":
          "IPAddress": "10.88.0.3",
[root@podman containers]# podman inspect -l | grep IPAddress\":
          "IPAddress": "10.88.0.4",
[root@podman containers]# podman ps
CONTAINER ID  IMAGE                           COMMAND         CREATED        STATUS            POR
S                NAMES
82d6075767be  docker.io/library/httpd:latest  httpd-foreground  4 minutes ago   Up 4 minute
 ago   0.0.0.0:80->80/tcp      nostalgic_hypatia
845ff52751b4  docker.io/library/httpd:latest  httpd-foreground  20 minutes ago  Up 20 minut
s ago  0.0.0.0:8080->8080/tcp  eager_ganguly
```

4.2.5 备份迁移

podman会先把容器打包成一个gz包，然后可以到远程服务器上导入。不知道是镜像小还是什么原因速度很快。感觉比docker导入导出快多啦，不知道是不是错觉。

```
[root@podman containers]# podman container checkpoint 82d6075767be -e /tmp/checkpoin
.tar.gz
82d6075767beff34a168ca48e47c5155bcd714eac1a53054049fbb841535832e
[root@podman containers]# ll /tmp/
总用量 1040
srwxr-xr-x 1 root root       0 12月  6 19:51 Aegis-<Guid(5A2C30A2-A87D-490A-9281-6765ED
D7CBA)>
-rw------- 1 root root 1058827 12月  6 21:01 checkpoint.tar.gz
drwx------ 3 root root    4096 12月  6 20:10 systemd-private-1254221ee241484f88bf1a3a3b2f
29b-chronyd.service-i0zmSw
[root@podman containers]# podman rm -f 82d6075767be
82d6075767beff34a168ca48e47c5155bcd714eac1a53054049fbb841535832e
[root@podman containers]# podman ps
CONTAINER ID  IMAGE                           COMMAND         CREATED        STATUS            POR
S                NAMES
845ff52751b4  docker.io/library/httpd:latest  httpd-foreground  25 minutes ago  Up 25 minut
s ago  0.0.0.0:8080->8080/tcp  eager_ganguly
[root@podman containers]# podman container restore -i /tmp/checkpoint.tar.gz
82d6075767beff34a168ca48e47c5155bcd714eac1a53054049fbb841535832e
[root@podman containers]# curl 127.0.0.1
<html><body><h1>It works!</h1></body></html>
[root@podman containers]#
```