containerd 入门

author: Uncle Dragon
date: 2021-06-16

安装之前

```
$ cat <<EOF | sudo tee /etc/modules-load.d/containerd.conf
overlay
br_netfilter
EOF

$ sudo modprobe overlay
$ sudo modprobe br_netfilter


# 设置必需的 sysctl 参数，这些参数在重新启动后仍然存在。
$ cat <<EOF | sudo tee /etc/sysctl.d/99-kubernetes-cri.conf
net.bridge.bridge-nf-call-iptables  = 1
net.ipv4.ip_forward                 = 1
net.bridge.bridge-nf-call-ip6tables = 1
EOF

# 应用 sysctl 参数而无需重新启动
$ sudo sysctl --system

```

安装 containerd

1. 从官方Docker仓库安装

参考 docker的安装
你没看错 就是 `docker` 的安装文档，我们只安装 `containerd.io` 就好

```
$ sudo apt-get update

$ sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg \
    lsb-release

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

$ echo \
  "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

```

```
$ sudo apt-get update

$ sudo apt-get install containerd.io
```

2. 下载 containerd 二进制文件，手动安装

```
# 下载链接
$ wget https://github.com/containerd/containerd/releases/download/v1.5.2/containerd-1.5.2-linux-amd64.tar.gz

# 解压二进制包并生成默认文件
$ tar xvf containerd-1.5.2-linux-amd64.tar.gz -C /usr/local/

# 创建目录
$ mkdir /etc/containerd

# 在创建的目录中生成配置文件
$ containerd config default> /etc/containerd/config.toml
```

配置 `containerd` 服务

```
$ sudo cat <<EOF | sudo tee /lib/systemd/system/containerd.service
> [Unit]
> Description=containerd container runtime
> Documentation=https://containerd.io
> After=network.target local-fs.target
>
> [Service]
> ExecStartPre=-/sbin/modprobe overlay
> ExecStart=/usr/local/bin/containerd
>
> Type=notify
> Delegate=yes
> KillMode=process
> Restart=always
> RestartSec=5
>
> LimitNPROC=infinity
> LimitCORE=infinity
> LimitNOFILE=1048576
>
> TasksMax=infinity
```

```
> OOMScoreAdjust=-999
>
> [Install]
> WantedBy=multi-user.target
> EOF
```

配置服务开机自启

```
$ systemctl enable containerd

$ systemctl start containerd

$ systemctl status containerd
```

配置

```
$ sudo mkdir -p /etc/containerd
$ containerd config default | sudo tee /etc/containerd/config.toml
```

结合 `runc` 使用 `systemd` cgroup 驱动，在 `/etc/containerd/config.toml` 中设置

```
[plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc]
  ...

[plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc.options]
    SystemdCgroup = true
```

配置容器镜像加速

编辑 `/etc/containerd/config.toml`

```
    [plugins."io.containerd.grpc.v1.cri".registry]
      [plugins."io.containerd.grpc.v1.cri".registry.mirrors]

[plugins."io.containerd.grpc.v1.cri".registry.mirrors."docker.io"]
          endpoint = ["https://dr6tjot4.mirror.aliyuncs.com"]
```

重新启动 containerd

```
$ sudo systemctl restart containerd
```

测试

```
$ ctr version
Client:
  Version:  1.4.6
  Revision: d71fcd7d8303cbf684402823e425e9dd2e99285d
  Go version: go1.13.15

Server:
  Version:  1.4.6
  Revision: d71fcd7d8303cbf684402823e425e9dd2e99285d
  UUID: 2ba390eb-5ebf-49ae-a931-fa3ec68bb8aa

```

使用

- `ctr` 是 `containerd` 本身的车重
- `crictl` 是 k8s 社区定义的专门的 CLI 工具
- `nerdctl` 是 containerd 推出了一个新的 CLI 工具

`ctr`

输出版本信息

```
# ctr version
Client:
  Version:  1.4.6
  Revision: d71fcd7d8303cbf684402823e425e9dd2e99285d
  Go version: go1.13.15

Server:
  Version:  1.4.6
  Revision: d71fcd7d8303cbf684402823e425e9dd2e99285d
  UUID: 2ba390eb-5ebf-49ae-a931-fa3ec68bb8aa
```

输出帮助信息

```
# ctr --help
NAME:
   ctr -
        __
  _____/ /______
 / ___/ __/ ___/
/ /__/ /_/ /
\___/\__/_/

containerd CLI

```

```
USAGE:
   ctr [global options] command [command options] [arguments...]

VERSION:
   1.4.6

DESCRIPTION:

ctr is an unsupported debug and administrative client for interacting
with the containerd daemon. Because it is unsupported, the commands,
options, and operations are not guaranteed to be backward compatible or
stable from release to release of the containerd project.

COMMANDS:
   plugins, plugin            provides information about containerd plugins
   version                    print the client and server versions
   containers, c, container   manage containers
   content                    manage content
   events, event              display containerd events
   images, image, i           manage images
   leases                     manage leases
   namespaces, namespace, ns  manage namespaces
   pprof                      provide golang pprof outputs for containerd
   run                        run a container
   snapshots, snapshot        manage snapshots
   tasks, t, task             manage tasks
   install                    install a new package
   oci                        OCI tools
   shim                       interact with a shim directly
   help, h                    Shows a list of commands or help for one command

GLOBAL OPTIONS:
   --debug                      enable debug output in logs
   --address value, -a value    address for containerd's GRPC server (default: "/run/containerd/containerd.sock") [$CONTAINERD_ADDRESS]
   --timeout value              total timeout for ctr commands (default: 0s)
   --connect-timeout value      timeout for connecting to containerd (default: 0s)
```

```
   --namespace value, -n value  namespace to use with commands (default: "default") [$CONTAINERD_NAMESPACE]
   --help, -h                   show help
   --version, -v                print the version

```

拉取镜像

```
# ctr images  pull docker.io/library/redis:latest
docker.io/library/redis:latest:
                resolved
|++++++++++++++++++++++++++++++++++++++|
index-sha256:7e2c6181ad5c425443b56c7c73a9cd6df24a122345847d1ea9bb86a5afc76325:    done           |++++++++++++++++++++++++++++++++++++++|
manifest-sha256:c2cbe8a592927bb74033f9c29b103ebc8e1ab3ed9598a9e937aaa2a723d5b8a7: done           |++++++++++++++++++++++++++++++++++++++|
config-sha256:fad0ee7e917aeec77f15d0a106b8e415f4c0499d341b88c841b9a9f78c3c3ca5:   done           |++++++++++++++++++++++++++++++++++++++|
layer-sha256:fa57f005a60dc23920d7e5fa67ca618938a4b57254a773b174f0fd4950e02258:    done           |++++++++++++++++++++++++++++++++++++++|
layer-sha256:bcdf6fddc3bdaab696860eb0f4846895c53a3192c9d7bf8d2275770ea8073532:    done           |++++++++++++++++++++++++++++++++++++++|
layer-sha256:2902e41faefa618f0e57a9e63a2827e5024243af5933ca3e3af8cacb2e91d98b:    done           |++++++++++++++++++++++++++++++++++++++|
layer-sha256:69692152171afee1fd341febc390747cfca2ff302f2881d8b394e786af605696:    done           |++++++++++++++++++++++++++++++++++++++|
layer-sha256:df3e1d63cdb17d09cdccce0ee473223fc6981b524573359d620b31a022e42b0b:    done           |++++++++++++++++++++++++++++++++++++++|
layer-sha256:a4a46f2fd7e06fab84b4e78eb2d1b6d007351017f9b18dbeeef1a9e7cf194e00:    done           |++++++++++++++++++++++++++++++++++++++|
elapsed: 17.3s
                total:  36.9 M (2.1 MiB/s)

unpacking linux/amd64 sha256:7e2c6181ad5c425443b56c7c73a9cd6df24a122345847d1ea9bb86a5afc76325...
done

```

查看镜像文件列表

```
# ctr images list
REF                                                             TYPE
                                                DIGEST
                                                    SIZE      PLATFORMS


                        LABELS
docker.io/library/nginx:alpine
application/vnd.docker.distribution.manifest.list.v2+json
sha256:6d76a25a64f6a9a873bded796761bf7a1d18367570281d73d16750ce37fa
e297 9.4 MiB
linux/386,linux/amd64,linux/arm/v6,linux/arm/v7,linux/arm64/v8,linu
x/ppc64le,linux/s390x                   -
docker.io/library/redis:latest
application/vnd.docker.distribution.manifest.list.v2+json
sha256:7e2c6181ad5c425443b56c7c73a9cd6df24a122345847d1ea9bb86a5afc7
6325 36.9 MiB
linux/386,linux/amd64,linux/arm/v5,linux/arm/v7,linux/arm64/v8,linu
x/mips64le,linux/ppc64le,linux/s390x -
repository.anxinyun.cn/base-images/alpine:latest
application/vnd.docker.distribution.manifest.v2+json
sha256:cc77c137c4e3fcaacb7f19f8b1582c0138f943dbbc27421060771ebb49e8
7a3a 7.2 MiB  linux/amd64
                                                        -
```

运行和查看容器

```
# ctr run -d docker.io/library/redis:latest my-redis
# ctr container list
CONTAINER    IMAGE                                RUNTIME

my-nginx     docker.io/library/nginx:alpine
io.containerd.runc.v2
my-redis     docker.io/library/redis:latest
io.containerd.runc.v2
```

删除容器

```
# ctr container list
CONTAINER    IMAGE                                RUNTIME

my-nginx     docker.io/library/nginx:alpine
io.containerd.runc.v2
my-redis     docker.io/library/redis:latest
io.containerd.runc.v2
# ctr task kill my-nginx

# ctr container list
```

```
CONTAINER    IMAGE                                    RUNTIME

my-nginx     docker.io/library/nginx:alpine
io.containerd.runc.v2
my-redis     docker.io/library/redis:latest
io.containerd.runc.v2
# ctr container del my-nginx

# ctr container list
CONTAINER    IMAGE                                    RUNTIME

my-redis     docker.io/library/redis:latest
io.containerd.runc.v2
```

ctr 没有 stop 容器的功能，只能暂停或者杀死容器

`crictl`

安装

- 使用 wget

```
VERSION="v1.21.0"
wget https://github.com/kubernetes-sigs/cri-
tools/releases/download/$VERSION/crictl-$VERSION-linux-amd64.tar.gz
sudo tar zxvf crictl-$VERSION-linux-amd64.tar.gz -C /usr/local/bin
rm -f crictl-$VERSION-linux-amd64.tar.gz
```

- 使用 curl

```
VERSION="v1.21.0"
curl -L https://github.com/kubernetes-sigs/cri-
tools/releases/download/$VERSION/crictl-${VERSION}-linux-
amd64.tar.gz --output crictl-${VERSION}-linux-amd64.tar.gz
sudo tar zxvf crictl-$VERSION-linux-amd64.tar.gz -C /usr/local/bin
rm -f crictl-$VERSION-linux-amd64.tar.gz
```

使用

这个是为了 k8s 使用containerd 而做的，后面配合k8s 再做具体的说明

下面是docker 和 containerd 的cli 工具使用常用命令对比

id	docker	ctr	crictl	备注
1	docker images	ctr images ls	crictl images	查看本地镜像
2	docker pull	ctr images pull	crictl pull	拉取镜像
3	docker run	ctr container run	-	运行容器
4	docker ps	ctr task ls	crictl ps	查看运行的人容器
5	docker rm	ctr container del	crictl rm	移除容器
6	docker exec	ctr task exec	crictl exec	进入容器
7	docker logs		crictl logs	查看容器日志

`nerdctl`

安装

他有两个版本，一个full 版的，一个mini 版

full 版本包含了依赖，像 containerd, runc, CNI 等

minimal :

```
wget https://github.com/containerd/nerdctl/releases/download/v0.8.3/nerdctl-0.8.3-linux-amd64.tar.gz
tar Cxzvvf /usr/local/bin nerdctl-0.8.3-linux-amd64.tar.gz
```

full :

```
wget https://github.com/containerd/nerdctl/releases/download/v0.8.3/nerdctl-full-0.8.3-linux-amd64.tar.gz
tar Cxzvvf /usr/local nerdctl-full-0.8.3-linux-amd64.tar.gz
```

如果之前安装了container 就可以直接安装 minimal 版的。

使用

使用上基本与 docker 一致，把 `docker` 换成 `nerdctl` 命令基本上都可以使用，

但是如果把`nerdctl` 简单的看作是 `docker` cli 的复制就大错特错了，他还实现了docker 不具备的功能，例如延迟拉取镜像（lazy-pulling）、镜像加密（imgcrypt）等

```
# nerdctl ps
```

```
CONTAINER ID    IMAGE                                    COMMAND
          CREATED        STATUS    PORTS    NAMES
my-redis        docker.io/library/redis:latest    "docker-
entrypoint.s…"    4 hours ago    Up

# nerdctl images
REPOSITORY                                       TAG       IMAGE ID
     CREATED        SIZE
busybox                                          latest
930490f97e5b    3 hours ago    1.3 MiB
nginx                                            alpine
6d76a25a64f6    5 hours ago    16.0 KiB
redis                                            latest
7e2c6181ad5c    5 hours ago    20.0 KiB
repository.anxinyun.cn/base-images/alpine    latest
cc77c137c4e3    3 days ago     5.3 MiB

# nerdctl exec -it my-redis sh
#

```