容器

1 容器技术介绍

容器是一种轻量级、可移植、自包含的软件打包技术，使应用程序可以在几乎任何地方以相同的方式运行。无需任何修改就能够在生产系统的虚拟机、物理服务器或公有云主机上运行。

1.1 容器的特点

跨环境、可移植、资源和应用隔离性、安全性

1.2 容器与虚拟机区别

•容器直接运行在内核上

•容器优势启动快、高性能和低延迟

•虚拟机可以虚拟硬件不依赖内核、隔离更加彻底

1.3 容器引擎

Singularity是目前在高性能计算平台上被大量应用的轻量虚拟化容器技术，能够提供操作系统级的虚拟化。

•Singularity 更轻、适合HPC，MPI(OpenMPI, MPICH, IntelMPI),GPU,infiniband

•Singularity 的用户权限容器内外都一致, /dev, /sys and /proc automount，安全性高

2 hep_container

Hep_container是基于singularity容器管理命令开发的适用于高能所计算集群的容器客户端工具，满足用户使用多种操作系统版本及环境的需求。 **说明：**本文涉及的命令均需要登陆节点上运行，所用命令在以下目录，建议将该目录加入用户个人环境变量 PATH 中。

```
/cvmfs/container.ihep.ac.cn/bin/
export PATH=$PATH:/cvmfs/container.ihep.ac.cn/bin/
```

Hep_container的特点

•hep_container容器统一运行入口程序

•用户对容器操作统一

•容器的参数配置和镜像路径对用户透明

•容器更新不会影响到当前用户的使用

•实现同一容器在不同站点上运行不同的作业配置策略，高能所、科大、北京大学等站点多种作业调度配置

•容器内用户只有自己实验组的数据盘访问权限，安全性高

2.1 命令说明

Hep_container的容器命令主要有以下操作images、shell、exec等。可以在命令行中通过help参数查看各个命令的使用说明和样例

```
$ hep_container  help
Usage : ./hep_container  <command> [command options...]
CONTAINER USAGE COMMANDS:
    shell           Run a Bourne shell within container image
    exec            Execute a command within container image
    images          List Support container images
    groups          List Support groups
    -g groupname    With a specific group name
EXAMPLES:
    ./hep_container images
    ./hep_container groups

    ./hep_container shell SL5
    ./hep_container shell SL5 -g physics

    ./hep_container exec  SL5 cat /etc/redhat-release
    ./hep_container exec  SL5 python ./yourprograme.py
    ./hep_container exec  SL5 -g physics cat /etc/redhat-release
```

2.2 查看支持镜像

命令格式：hep_container images 该指令可以查看当前提供的操作系统容器镜像。

```
$ hep_container images
Hep_container support images:
        SL5 : Scientific Linux 5
        SL6 : Scientific Linux 6
        SL7 : Scientific Linux 7
```

2.3 查看支持用户组/实验组

命令格式：hep_container groups 该指令可以查看容器命令当前支持提供的用户组或者实验组。通过 -g 参数指定用户组或实验组，容器内会挂载对应用户目录和实验目录。不指定-g参数默认采用主组作为用户组或实验组。

```
$ hep_container groups
Hep_container support groups:

u07|atlas|atlasrun|comet|offline|physics|higgs|ams|cms|dyw|hxmt|polars|juno|argo
|lhaaso|sch
```

2.4 进入容器环境

命令格式：hep_container shell [container image] 该指令可以在容器内启动一个shell，因此可以在容器外部与容器内部进行交互操作。运行exit则可以退出该shell。 下例为运行启动一个SL5操作系统镜像后，用户当前为SL5的系统环境.

```
$ hep_container shell SL5
Singularity: Invoking an interactive shell within container...
Singularity> cat /etc/redhat-release
Scientific Linux SL release 5.5 (Boron)
Singularity> exit
exit
```

2.5 容器内执行命令

命令格式：hep_container exec [container image] [command] 该指令可以在外部主机上将指定的command运行在指定的容器内。 下例为在lxslc7上以SL5的环境运行SL5命令，并得到结果。

```
$ hep_container exec SL5 cat /etc/redhat-release
Scientific Linux SL release 5.5 (Boron)
```

3 镜像制作

3.1 通过定义文件制作镜像

```
[user1@dockertest02]# cat mymkimage.def
BootStrap:yum
OSVersion: 7.8

   MirrorURL: http://mirror.ihep.ac.cn/centos/7/os/x86_64/

   UpdateURL: http://mirror.ihep.ac.cn/centos/7/os/x86_64/

   Include: yum

%setup

%files

   #~/home/yourfile  ~/usr/local/yourfile

%runscript
```

```
  echo "Running the container..."

%post

    echo "Installing base group"
    yum -y groupinstall "Minimal Install"
    echo "Installing basic packages"
    yum -y install vim-enhanced wget ntp gcc gcc-c++ glibc make
    echo "Installing requied packages"
    yum -y install python36
```

```
sudo singularity build mymkimage.sif mymkimage.def
```

3.2 使用自制镜像

```
export MYIMAGE=/home/sch/test001/mymkimage.sif
hep_container shell MYIMAGE
```

3.3 其他镜像制作方法

参考 https://sylabs.io/guides/3.5/user-guide/build_a_container.html