Singularity 利用手順書
(AOBA-S 環境)

2024 年 5 月

第一版

＜＜　改版履歴　＞＞

版数	改版日	改版理由	改版者
第一版	2024.5	第一版として作成	

目次

本資料について 2
1　コンテナ作成手順 3
1.1　VE 用コンテナの作成 3
1.2　コンテナのカスタマイズ 5
2　VE 用コンテナを使用した Singularity の実行 6
2.1　バッチジョブとしての実行方法 6
2.2　インタラクティブジョブとしての実行方法 7
3　制限事項 8

本資料について

本資料は，Singularity で AOBA-S の VE 用コンテナを作成し，利用するための手順を記載したものです.

1　コンテナ作成手順

1.1　VE 用コンテナの作成

AOBA-S の VE に対応した Singularity コンテナを作成します.

Github で公開されている VEOS コンテナのレシピを使用します.

コンテナの作成は AOBA-S のフロントエンドサーバで実施します.

参考：

https://github.com/veos-sxarr-NEC/singularity

例）${HOME}/work 配下に作成する場合

```
$ mkdir ${HOME}/work
$ cd ${HOME}/work
$ git clone https://github.com/veos-sxarr-NEC/singularity.git
$ cd ${HOME}/work/singularity/RockyLinux8
$ curl -O https://sxauroratsubasa.sakura.ne.jp/repos/TSUBASA-soft-release-ve3-3.0-1.noarch.rpm

リポジトリファイル内の下線部を修正し，AOBA-S 内のリポジトリサーバ(172.21.0.84)から VEOS ソフトウェアをダウンロードするよう変更します.

$ cp -p TSUBASA-repo.repo TSUBASA-repo.repo.org
$ vi TSUBASA-repo.repo
baseurl=https://sxauroratsubasa.sakura.ne.jp/repos/TSUBASA-repo_el8.8
      ↓
baseurl=http://172.21.0.84/repos/TSUBASA-repo_el8.8

baseurl=https://sxauroratsubasa.sakura.ne.jp/repos/runtime/sdk/sdk_el8
      ↓
baseurl=http://172.21.0.84/repos/sdk_el8

baseurl=https://sxauroratsubasa.sakura.ne.jp/repos/runtime/mpi/mpi_mofed5_el8
      ↓
baseurl=http://172.21.0.84/repos/mpi_mofed5_el8

baseurl=https://sxauroratsubasa.sakura.ne.jp/repos/community/sdk/sdk_el8
      ↓
baseurl=http://172.21.0.84/repos/sdk_el8

baseurl=https://sxauroratsubasa.sakura.ne.jp/repos/community/mpi/mpi_mofed5_el8
      ↓
baseurl=http://172.21.0.84/repos/mpi_mofed5_el8
```

singularity コマンド実行前に，ユーザリソース(仮想メモリサイズ)を最大値に変更します.

```
$ ulimit -v
8388608
$ ulimit -H -v
67108864
$ ulimit -v 67108864
$ ulimit -v
67108864
```

module コマンドで singularity をロードし，build コマンドでコンテナを作成します.

```
$ module load Singularity/3.11
$ module list
Currently Loaded Modulefiles:
  1) ncc/5.2.0(default)        3) necmpi/3.6.0(default)     5) Singularity/3.11
  2) nfort/5.2.0(default)      4) nlc/3.1.0(default)
$ which singularity
/mnt/lustre/ap/singularity/3.11/bin/singularity

$ singularity build --fakeroot veos.sif veos.def
INFO:    Starting build...
   :
INFO:    Creating SIF file...
INFO:    Build complete: veos.sif
```

1.2　コンテナのカスタマイズ

任意のソフトウェア実行環境を含んだコンテナを作成するときは，1.1 で取得した veos コンテナ環境のレシピファイル veos.def を修正し，コンテナ内に必要なソフトウェアのインストール指示，および実行時の環境変数を設定します.

レシピファイル作成後，1.1 の手順と同様に，singularity build コマンドでコンテナファイルを作成します.

例）VEOS 用コンテナレシピファイル

```
Bootstrap: docker
From: rockylinux:8.8.20230518
%files
        dnf.conf /etc/dnf
        TSUBASA-soft-release-*.noarch.rpm /etc/yum.repos.d
        TSUBASA-repo.repo /etc/yum.repos.d
        TSUBASA-restricted.repo /etc/yum.repos.d

%post
        mv /etc/yum.repos.d/TSUBASA-repo.repo /etc/yum.repos.d/TSUBASA-repo.repo.bk
        mv /etc/yum.repos.d/TSUBASA-restricted.repo /etc/yum.repos.d/TSUBASA-restricted.repo.bk
        yum -y install /etc/yum.repos.d/TSUBASA-soft-release-*.noarch.rpm
        mv /etc/yum.repos.d/TSUBASA-repo.repo.bk /etc/yum.repos.d/TSUBASA-repo.repo
        mv /etc/yum.repos.d/TSUBASA-restricted.repo.bk /etc/yum.repos.d/TSUBASA-restricted.repo
        yum clean all
        yum -y group install ve-container nec-sdk-runtime -d 10
        sed -i -e "s|username=.*|username=|" -e "s|password=.*|password=|" ¥
            /etc/yum.repos.d/TSUBASA-restricted.repo
```

説明：

Bootstrap: , From:　　・・・　ベースイメージの種類，場所

%files　　・・・　ホスト OS からコンテナにコピーしたいファイル

%post　　・・・　イメージの準備後にコンテナ内で実行するコマンドを記述

他の項目については，以下のオンラインマニュアルをご参照ください.

Definition Files — SingularityCE User Guide 3.11 documentation (sylabs.io)

https://docs.sylabs.io/guides/3.11/user-guide/definition_files.html

2　VE 用コンテナを使用した Singularity の実行

NQSV を使用し，AOBA-S 上で Singularity を実行します.

実行方法としては，NQSV バッチシステムを使用して，バッチジョブとして Singularity exec モードで実行する方法と，インタラクティブジョブとして SX-Aurora サーバにログインし，Singularity shell モードで実行する方法の２つがあります.

2.1　バッチジョブとしての実行方法

バッチジョブとして Singularity exec モードでプログラムの実行する方法を以下に記載します.

フロントエンドサーバ上で，ジョブスクリプトを作成し，それをジョブ管理システム(NQSV)に投入することで，ジョブが計算ノード上で実行されます.

以下にジョブスクリプト（job.sh）の記述例を記載します.

※ 利用者のログイン環境によっては module の load に失敗することがあります. 投入用スクリプト内で「module load ～」行の前に，以下を記載することで回避できます.

source /etc/profile.d/modules.sh（投入用スクリプトが sh or bash の場合）

source /etc/profile.d/modules.csh（投入用スクリプトが csh の場合）

※ 実行プログラム(a.out)はあらかじめ VE 用 NEC コンパイラで作成し，ジョブを投入したディレクトリに格納しておきます.

```
#!/bin/sh
#PBS -q sxs
#PBS -l elapstim_req=00:10:00
#PBS --venode 8

module load Singularity/3.11

cd $PBS_O_WORKDIR
singularity exec --bind /var/opt/nec/ve/veos ./veos.sif ./a.out
```

上記で作成したジョブスクリプトをバッチジョブとして実行する場合，フロントエンドサーバ上で qsub コマンドを使用します.

例えば，ジョブスクリプト（job.sh）を投入する場合，以下のようにします.

実行結果は，ジョブを投入したディレクトリ上に出力される標準出力ファイルに返

されます.

```
$ qsub job.sh
```

2.2　インタラクティブジョブとしての実行方法

インタラクティブジョブとして Singularity shell モードでプログラムを実行する方法を以下に記載します.

フロントエンドサーバ上で，qlogin コマンドを実行してインタラクティブセッションが開始した後，VEOS 用コンテナを指定して singularity を起動したうえで，プログラムを実行します.

```
$ qlogin -q inter
プロジェクトコード AAAA リクエストを投入します
Request XXXXX.sjob submitted to queue: inter.
Waiting for XXXXX.sjob to start.
[sxat3001 ~]$ module load Singularity/3.11
[sxat3001 ~]$ module list
Currently Loaded Modulefiles:
 1) ncc/5.2.0(default)       3) necmpi/3.6.0(default)    5) Singularity/3.11
 2) nfort/5.2.0(default)     4) nlc/3.1.0(default)
$ singularity shell --bind /var/opt/nec/ve/veos veos.sif
Singularity> cd ~/test
Singularity> ./a.out
※プログラム実行時の標準出力・標準エラー出力が出力されます.
Singularity> exit
exit
[sxat3001 ~]$ exit
ログアウト
$
```

3　制限事項

VE 用のコンテナの作成・ジョブ実行するにあたり，以下の制限事項があります.

１．作成したコンテナは，SDK runtime のみ使用可能です.

２．AOBA-S 上では，--fakeroot オプションはエラーとなりますので，コンテナの作成はフロントエンドサーバで実施してください.

３．作成したコンテナにソフトウェアパッケージを追加したい場合，SANDBOX 形式への変換は，ローカル上に大量のファイルの出力が発生するため推奨しません. 時間も非常にかかるため，レシピファイルを作成しコンテナを再 build する方法を強く推奨します.

４．Singularity で実行されたコンテナファイルのファイルシステムはリードオンリーでマウントされるため，コンテナ内の OS に対してパッケージの追加インストールはできません.

以上