Singularity

Luis Concha

Qué es Singularity

- Comprime software en un solo archivo
- El archivo (contenedor) tiene TODO el sistema operativo que soporta al software.
- Incluye todas las librerías.
- Aumenta la reproducibilidad

Ventajas

- Permite correr software no instalado en máquina huésped.
- No se requiere ser super usuario en máquina huésped
- Fácil: No se requiere ser super-usuario para correr un contenedor.
- Seguro: Es imposible escalar privilegios del contenedor al huésped).

Instalación

- No usar `apt`, pues instalará versión vieja.
- Seguir pasos de instalación.
- No es difícil.

Contenedores pre-fabricados

- búsqueda de contenedores:

```
(base) lconcha@syphon:~/singularity$ singularity search dmri
No users found for 'dmri'

No collections found for 'dmri'

Found 1 containers for 'dmri'
    library://guillaumeth/default/dmriqc
        Tags: latest
```

- También se pueden buscar contenedores en singularity-hub
- Y otra manera más es convertir contenedores de `docker` a `singularity`.

Generar un contenedor que encontramos

`singularity build mycontainer_dmriqc.sif library://guillaumeth`

- Los archivos de los contenedores pueden ser de varios GB.

:warning: `build` y `pull` hacen aproximadamente lo mismo, pero `pull` pone el contenedor en el *cache* (habitualmente en `$HOME/.singularity/cache`), mientras que `build` nos deja generar un archivo `.sif`. - Esto es relevante en el clúster, donde nuestro `$HOME` no tiene mucho espacio. - Hacer archivos `.sif` aumenta la reproducibilidad. - Podemos convertir a `.sif` un container que estaba en el *cache* usando `build`.

Generando un container desde cero

- Hay que tener un archivo de definición `.def`, que contiene la *receta* a seguir para generar un contenedor. Ahí se pone lo que vamos a instalar y toda la configuración.
- Un archivo ejemplo muy simple, llamado `my_ubuntu.def`:

```
Bootstrap: library
From: ubuntu:20.04
```

- Lo llamaríamos armar así:

```
sudo singularity build my_ubuntu.sif my_ubuntu.def
```

- Al archivo `.def` se le pueden poner muchas cosas y configuraciones, que vienen explicadas aquí.
- Podemos usar `build` con `--sandbox` para ir instalando poco a poco los componentes. Una vez generado el `.sif`, podemos entrar a él con `sudo singularity shell --writable my_ubuntu.def`, y podemos ir poniéndole cosas.
- Esto es útil cuando vamos generando nuestro archivo de definición `.def`

Corriendo el contenedor

- La mayoría de los contenedores tienen un `%runscript`, que se ejecuta automáticamente al correr el contenedor.
- El `%runscript` puede ser complejo y recibir muchos argumentos y switches, como pasa con el contenedor de `fmriprep`

- ... pero también puede ser algo tan sencillo como *corre el comando que te pidan*. Esto se logra con un `%runscript` muy simple: `exec "$@"`
- Algo muy padre con un runscript sencillo es que podemos usar comandos de dentro del contenedor, para hacerle cosas a nuestros datos de *fuera* del contenedor. Para ello, sin embargo es *crucial* hacer un `bind` de la carpeta de *fuera* del contenedor, para que se vea *adentro* . Es fácil. Por ejemplo, en un contenedor que incluye a mrtrix, usaremos el comando `mrstats` del contenedor para trabajar en un archivo de *fuera* del contenedor:

```
singularity run -B /mnt test.sif mrstats /mnt/part1/data/datos
```

En este ejemplo tenemos un archivo en una subcarpeta dentro de `/mnt` en la máquina host, pero lo hacemos visible dentro del contenedor `test.sif` mediante el switch `-B /mnt`.