Dockerizing MDSplus

Stephen Lane-Walsh, MDSplus Developer, MIT
slw@psfc.mit.edu

This research was funded under DOE cooperative agreement DE-SC0012470.

Background Knowledge

1. **Background Knowledge**
2. MDSplus Containers
3. How to Use Them
4. Demo
5. Benefits / Limitations

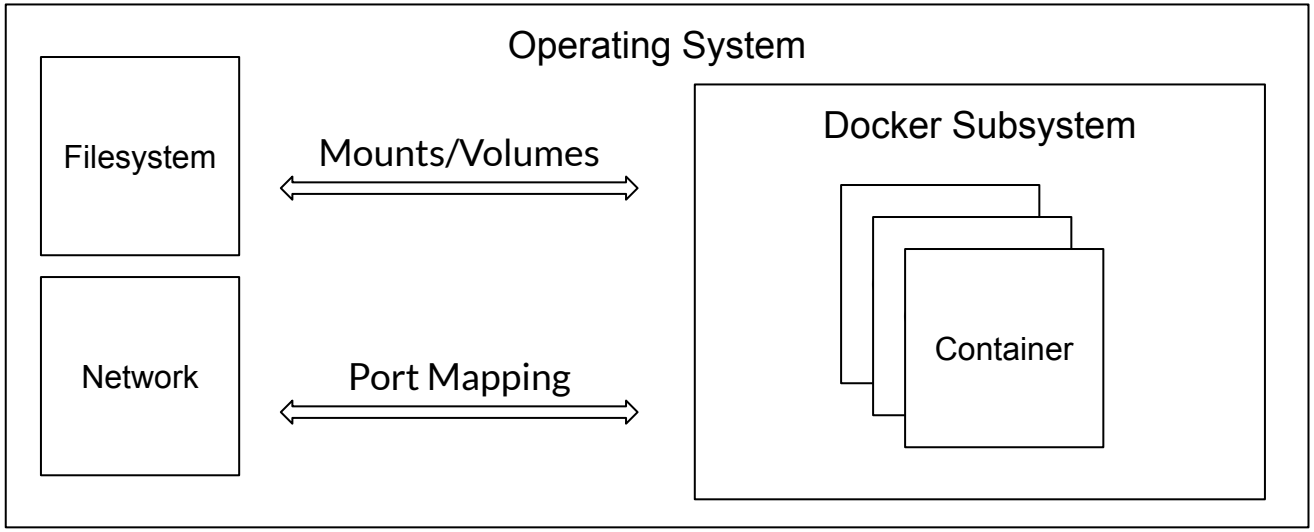
MDSplus Infrastructure

- MDSplus as a set of servers
- Can be difficult to install/configure
- Running on one machine or many
- Configuration dependent on constraints of experiment

Tree Server	Serve tree data files, evaluate expressions
DAQ Server	Run data acquisition scripts and communicate with devices
Dispatch Server	Build dispatch tables to ensure actions are run in order Control actions on DAQ server
Analysis Server	Compute data and store results

Docker

- Command line tool/service
- Software containers, can be viewed as “Lightweight VMs”
- Virtual network, port forwarding, host filesystem mounts
- Runs on any linux kernel
- Variants exist for Windows/OSX
- Share images on https://hub.docker.com



Docker Terminology

Images	Packaged filesystems, analogous to executables Can be shared or extended
Containers	Running instances of executables Multiple containers can run off of one image
Network	Virtual network in which all the containers have IPs, hostnames, etc
Volume	A mounted filesystem either shared between multiple containers, or between the container and the host filesystem

Docker Compose

- Define relationships between containers
- Deploy on a server or your computer
- Infrastructure as Code
- Easy to destroy/recreate
- Self contained
- Can add config for building images
- YAML

docker-compose.yml

```yaml
version: "3.3"
services:
  tree_server:
    image: "whobrokethebuild/mdsplus:tree-server"
    environment:
      - "demo_path=/trees/~t/"
      - "UID=${UID}"
      - "GID=${GID}"
    volumes:
      - ./trees:/trees
      - ./pydevices:/pydevices
      - ./scripts:/scripts
    ports:
      - "8000:8000"
  dispatch_server:
    image: "whobrokethebuild/mdsplus:mdsip-server"
    env_file:
      - trees.env
      - servers.env
    environment:
      - "MDSIP_PORT=8101"
      - "UID=${UID}"
      - "GID=${GID}"
    volumes:
      - ./pydevices:/pydevices
      - ./scripts:/scripts
    ports:
      - "8101:8101"
  daq_server:
    image: "whobrokethebuild/mdsplus:mdsip-server"
    env_file:
      - trees.env
      - servers.env
    environment:
      - "MDSIP_PORT=8102"
      - "UID=${UID}"
      - "GID=${GID}"
    volumes:
      - ./pydevices:/pydevices
      - ./scripts:/scripts
    ports:
      - "8102:8102"
  analysis_server:
    image: "whobrokethebuild/mdsplus:mdsip-server"
    env_file:
      - trees.env
      - servers.env
```

Docker Compose

docker-compose.yml

```yaml
version: "3.3"
services:
  tree_server:
    image: "whobrokethebuild/mdsplus:tree-server"
    environment:
      - "demo_path=/trees/~t/"
      - "UID=${UID}"
      - "GID=${GID}"
    volumes:
      - ./trees:/trees
      - ./pydevices:/pydevices
      - ./scripts:/scripts
    ports:
      - "8000:8000"
```

MDSplus Containers

1. Background Knowledge
2. **MDSplus Containers**
3. How to Use Them
4. Demo
5. Benefits / Limitations

MDSplus Images

- Alpine Linux based image
- Includes most commonly used packages
- Optimized for general use
- Minimal configuration
- Providing channel-defined and version-defined tags

mdsplus	General MDSplus installation Used for client tools Base package of **tree-server** and **mdsip**
tree-server	Specialized and configured to use inetd to spawn mdsip processes to handle connections
mdsip	Specialized and configured to run a single mdsip process on a specified port Used for dispatch, daq, and analysis

Container Configuration

- Environment variables for tree paths
- MDSIP_PORT for mdsip based servers
- Standard mounts for common needs

/trees/	Folder containing all tree data files e.g. **/trees/demo/**
/tdi/	Folder containing all tdi functions
/pydevices/	Folder containing all python device classes
/scripts/	Folder containing all of your scripts to run
/scopes/	Folder containing dwscope or jScope definition files

Extending the Images

- Use the MDSplus containers as base images
- Add your config, scripts, etc.
- Install your own software, python packages, etc.

```
FROM whobrokethebuild/mdsplus:alpha
LABEL maintainer="Stephen Lane-Walsh <slw@psfc.mit.edu>"

COPY mymdsplus.conf /etc/mdsplus.conf
COPY entrypoint.sh /entrypoint.sh

ENTRYPOINT ["/entrypoint.sh"]
```

How to Use Them

Portable Docker Compose

- Use my General images or Build with compose
- Create an architecture as a file
- Well defined
- Easy to update and redeploy
- Requires Docker + Docker Compose installed

General Containers + Systemd

- Use systemd (or any service manager)
- Use docker commands to create and maintain processes
- Requires more effort to install on a system
- Very robust
- Easy to update and restart services

Build Your Own Containers

- Build your own containers
- Either inherit from mine, or write your own
- The most configurable
- Advanced
- Can be used with docker-compose or systemd
  - docker-compose helps with building

Client Tools using the General Containers

- Use any tool packaged with MDSplus on your system
  - `mdstcl`, `dwscope`, `jScope`, `traverser`, `jTraverser`, `actmon`
- No installation needed
- Can connect to existing infrastructure

Demo

Demo

- Available at:
  https://github.com/WhoBrokeTheBuild/DockerizedMDSplus

  https://hub.docker.com/r/whobrokethebuild/mdsplus
- General Containers + Docker Compose
- Full shot cycle
- Helper script with docker function wrappers
- Come see a live demo at the MDSplus workshop

Demo

Start Servers

```
$ UID=$(id -u) GID=$(id -g) docker-compose up -d
Creating network "demo_default" with the default driver
Creating demo_daq_server_1       ... done
Creating demo_dispatch_server_1 ... done
Creating demo_analysis_server_1 ... done
Creating demo_tree_server_1      ... done
$
```

Source Helper Script

```
$ . ./setup.sh
non-network local connections being added to access control list
```

Create Tree

```
$ demo-mdstcl """@/scripts/create_demo_tree.tcl"""
$
```

docker-compose.yml

```yml
version: "3.3"
services:
  tree_server:
    image: "whobrokethebuild/mdsplus:tree-server"
    environment:
      - "demo_path=/trees/~t/"
      - "UID=${UID}"
      - "GID=${GID}"
    volumes:
      - ./trees:/trees
      - ./pydevices:/pydevices
      - ./scripts:/scripts
    ports:
      - "8000:8000"
  dispatch_server:
    image: "whobrokethebuild/mdsplus:mdsip-server"
    env_file:
      - trees.env
      - servers.env
    environment:
      - "MDSIP_PORT=8101"
      - "UID=${UID}"
      - "GID=${GID}"
    volumes:
      - ./pydevices:/pydevices
      - ./scripts:/scripts
    ports:
      - "8101:8101"
  daq_server:
    image: "whobrokethebuild/mdsplus:mdsip-server"
    env_file:
      - trees.env
      - servers.env
    environment:
      - "MDSIP_PORT=8102"
      - "UID=${UID}"
      - "GID=${GID}"
    volumes:
      - ./pydevices:/pydevices
      - ./scripts:/scripts
    ports:
      - "8102:8102"
  analysis_server:
    image: "whobrokethebuild/mdsplus:mdsip-server"
    env_file:
      - trees.env
      - servers.env
```

Demo

Bash or Python prompt

```
$ demo-shell
/ # python
Python 2.7.15 (default, Aug 22 2018, 13:28:29)
[GCC 6.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
/ # exit
```

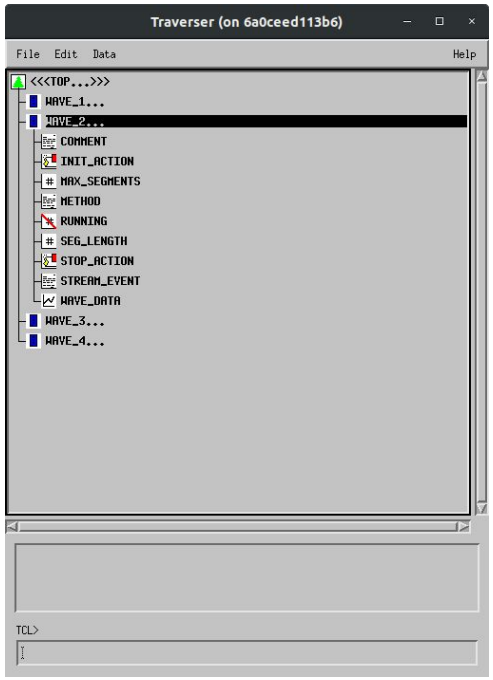
TCL prompt

```
$ demo-mdstcl
TCL> set tree demo
TCL> dir /full *

\DEMO::TOP

 :WAVE_1
      Status: on,parent is on, usage device,readonly
      compress on put
      Data inserted:  3-MAY-2019 21:10:01.89    Owner: gid=0(root),uid=0(root)
      Dtype: DTYPE_CONGLOM         Class: CLASS_R             Length: 157 bytes
      Model element: 1
      Device Help:
    WaveDevice class for DAQ testing
```

Demo

Start Traverser

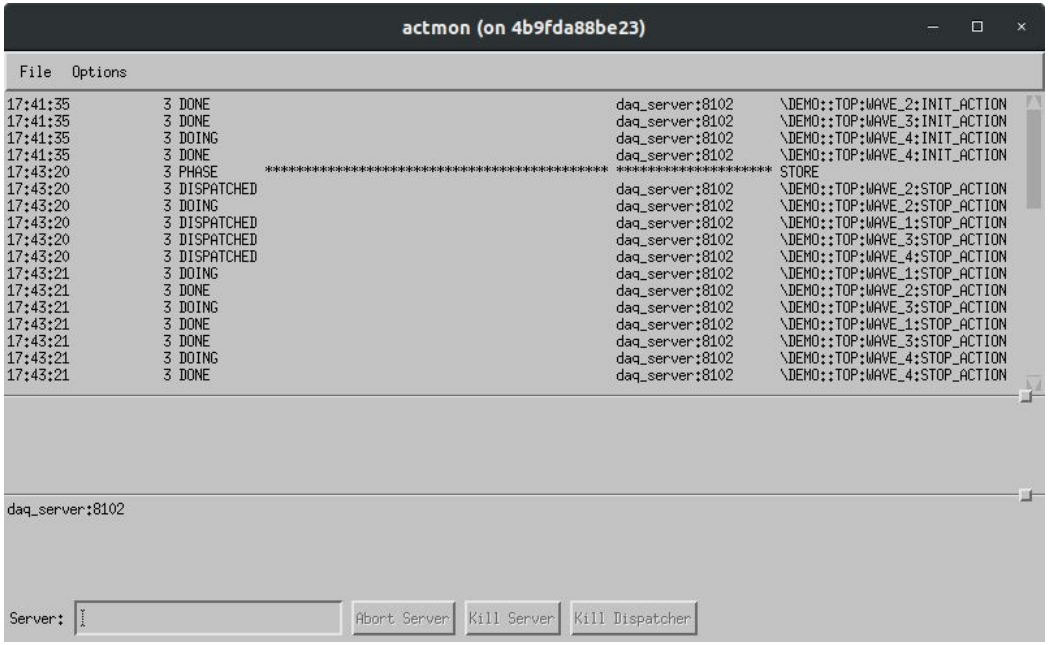
```
$ docker run -d --rm -it $(echo $DOCKER_FLAGS) --env=DISPLAY \
    --env=QT_X11_NO_MITSHM=1 --volume=/tmp/.X11-unix:/tmp/.X11-unix:rw \
    whobrokethebuild/mdsplus:latest traverser
28c818f883889a3be6fda9534d28460ffe42da113f59f27ccba770bedbf4e1a7

# Or run the bash function

$ demo-traverser
f2d907f1f888aac22f39ef6a0355386e77afd4a69c70d585d141405ae90e0055
```



Demo

Start Action Monitor

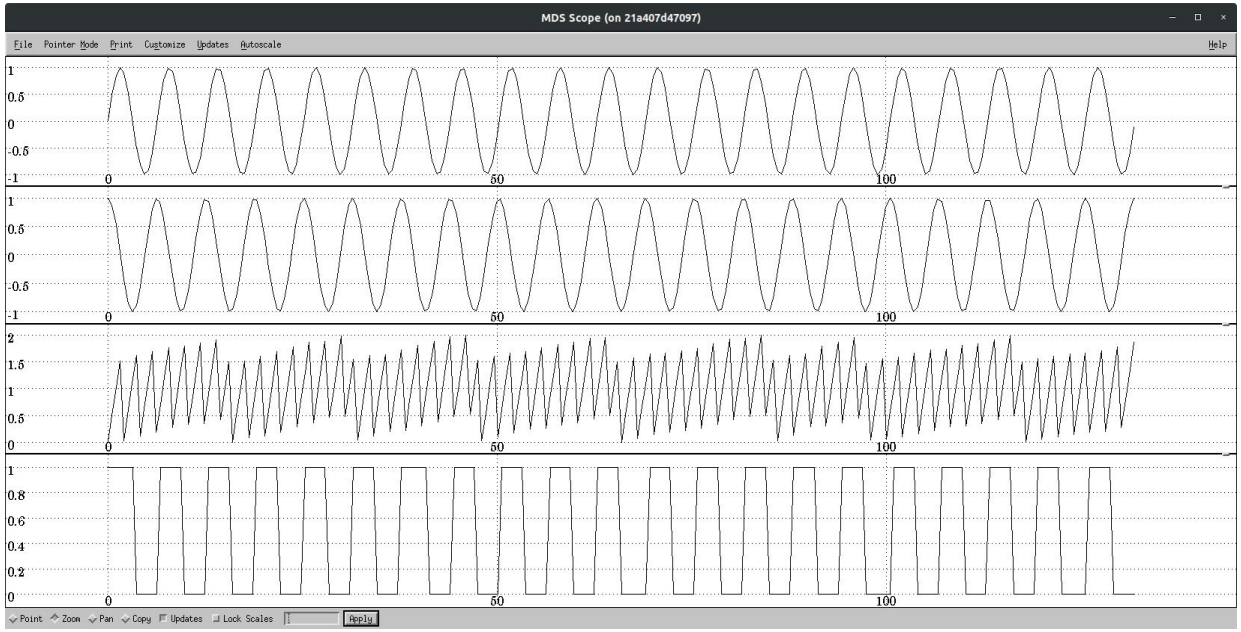
```
$ demo-actmon -tree demo -monitor event:demo_actmon
e726328b5994a6d06c4bb2de8a3302cedf5451033da42760e47d835bb928c86f
$
```



Demo

Start Scope

```
$ demo-mdstcl dispatch /command /server=dispatch_server:8101 """@/scripts/shot.tcl"""
$
$ demo-dwscope -def /scopes/wave.dat
e757f21361f634c9974001da28b613779c724a1a966c97f4ca771d5ec7ba5bc9
$
```



Benefits / Limitations

Benefits

- No MDSplus installation
  - Docker installation still needed
- Easy to upgrade
  - Change image tag from 7.63.1 to 7.63.2
- Infrastructure as code
  - Entire MDSplus infrastructure stored in docker-compose.yml or systemd

Limitations

- GUI applications and X-forwarding
  - Linux Only
  - OSX/Windows possible with effort
- UID/GID mapping
  - Docker runs as root
  - `UID=$(id -u) GID=$(id -g) docker-compose up -d`
- Performance
  - Little to no impact on server applications
  - Unknown impact on GUI applications
- Have to specify all `tree_path` environment variables
  - Fixed in 7.74.0 with `default_tree_path`

END