ICN EdgeXFoundry in docker-compose

This document will describe the process of launching the EdgeXFoundry services with docker-compose.

Prerequisites:

1. Docker-engine v17.06.0+ to support compose file format v3.4 for Fuji release
2. Docker-compose version 1.25+
    a. Unsure of minimal version, but docker-compose should be capable of using docker-compose file version 3.4 for fuji release of EdgeXFoundry
3. jq command line tool to use scripts below

Setup Process

1. Obtain the compose files for EdgeXFoundry
    a. `git clone https://github.com/edgexfoundry/developer-scripts`
2. Navigate to the compose files for the fuji release
    a. `cd developer-scripts/releases/fuji/compose-files`
3. Link the no-security variant to a docker-compose.yaml file for simple operation
    a. `ln -s ./docker-compose-fuji-no-secty.yml ./docker-compose.yml`
4. Edit the docker-compose.yml to *uncomment* the device-random section
    a. "device-random" section at about line 335-350
5. Ensure DOCKER_HOST is defined in the environment (docker-compose may fail to start if it is not defined)
    a. `export DOCKER_HOST=$HOSTNAME`
6. Start up the containers in background mode with docker-compose
    a. `sudo docker-compose up -d`

Tips for interacting with EdgeXFoundry via docker-compose & web UI

Description	Command (preface with sudo)
List services from docker-compose file	`docker-compose config --services`
View the logs of a particular service (add -f to follow)	`docker-compose logs [-f] <serviceName>`
view status of docker-compose containers (inside the compose-files folder)	`docker-compose ps`

Consul Web UI: http://localhost:8500/ui

EdgeXFoundry Web UI (admin/admin): http://localhost:4000

Script Samples for interacting with EdgeXFoundry

```
#!/bin/bash
# file: ping_edgex_services.sh
# description: ping all services in EdgeXFoundry containers
# reference: https://nexus.edgexfoundry.org/content/sites/docs/staging/master/docs/_build/html/Ch-GettingStartedUsers.html#checking-the-status-of-edgex-foundry
for PORT in {48095,48100,48082,48080,48081,49990,49988,27017,48061,48060,48075,48085,48090,4000}; do
    echo "Checking localhost:${PORT} ... Response: \"$(curl -s http://localhost:${PORT}/api/v1/ping )\""
done
```

```
#!/bin/bash
# file: get_random_data.sh
# description: script will use curl to pull random integers from the edgex-device-random service
# references:
# https://nexus.edgexfoundry.org/content/sites/docs/staging/master/docs/_build/html/Ch-
ExamplesVirtualDeviceService.html
# https://nexus.edgexfoundry.org/content/sites/docs/staging/master/docs/_build/html/Ch-QuickStart.
html#controlling-the-device

# pre: query the device-random service to get the IDs of the random-integer endpoints
RAND_INT_DEV="Random-Integer-Generator01"
RAND_INT_DEV_FILE="${RAND_INT_DEV}.json"
CORE_COMMAND_PORT=48082
curl -s http://localhost:${CORE_COMMAND_PORT}/api/v1/device/name/${RAND_INT_DEV} > $RAND_INT_DEV_FILE
RAND_INT_DEVICE_JSON=$(curl -s http://localhost:${CORE_COMMAND_PORT}/api/v1/device/name/${RAND_INT_DEV})

# CORE_DATA_PORT=48080

#for i in {0..2}; do
#    cat $RAND_INT_DEV_FILE | jq ".commands[${i}].get.path" | tr -d '"'
#    cat $RAND_INT_DEV_FILE | jq ".commands[${i}].get.url" | tr -d '"'
#done

# select URL for get commands
Int8_url=$( cat $RAND_INT_DEV_FILE | jq ".commands[0].get.url" | tr -d '"' | sed 's/edgex-core-command
/localhost/g' )
Int16_url=$(cat $RAND_INT_DEV_FILE | jq ".commands[1].get.url" | tr -d '"' | sed 's/edgex-core-command
/localhost/g' )
Int32_url=$(cat $RAND_INT_DEV_FILE | jq ".commands[2].get.url" | tr -d '"' | sed 's/edgex-core-command
/localhost/g' )

# query device for random integers
Int8_output=$( curl -s ${Int8_url})
Int16_output=$(curl -s ${Int16_url})
Int32_output=$(curl -s ${Int32_url})

# pipe the output through jq to select "value" fields
echo "Random Int8: $( echo $Int8_output | jq '.readings[0].value' | tr -d '"')"
echo "Random Int16: $(echo $Int16_output | jq '.readings[0].value' | tr -d '"')"
echo "Random Int32: $(echo $Int32_output | jq '.readings[0].value' | tr -d '"')"
```