Instructions for Installing Extractor Info Fetcher Service

This document is currently under development.

This document details the instructions for installing the Brown Dog auxiliary service *Extractor Info Fetcher* in BD-Clowder-dev (dev) and BD-Clowder (prod) machines.

1. SSH into to bd-clowder.ncsa.illinois.edu or bd-clowder-dev.ncsa.illinois.edu. Rest of the instructions are identical for both these machines unless specified otherwise.
2. Clone extractor-info-service code repository into /home/browndog
    a. `git clone opensource.ncsa.illinois.edu/bitbucket/scm/bd/bd-aux-services.git`

3. Install *pyvenv* if needed. One can also create Python virtual environment using *virtualenv*

    a. `sudo apt-get install python3-venv`

4. Install extractor-info-service

    a. `cd /home/browndog/bd-aux-services/extractor-info-fetcher`
    b. `sudo pyvenv prototype-endpoint-env`
    c. `source ./prototype-endpoint-env/bin/activate`

    d. `sudo -H ./prototype-endpoint-env/bin/pip install -r requirements.txt`

    e. Update *config.py* so that the variables, especially *rabbitmq_vhost*, *clowder_url, clowder_key* store details about development or production instances based on the machine (i.e. dev or prod) in which the service is being installed. I.e. Extractor Info Fetcher in bd-clowder machine should use information about Brown Dog production instances where as in bd-clowder-dev machine, it should use information about Brown Dog development instances.
5. Convert source code from Python2.7 to Python3 compatible by running (or something similar):

    a. `sudo 2to3-3.5 -w extractor-info-fetcher.py`
6. Update file permissions:
    a. `sudo chown -R browndog /home/browndog/bd-aux-services/`

    b. `sudo chgrp -R users /home/browndog/bd-aux-services/`
7. Make sure that the flask app can be run locally:

    a. `sh ./run-script.sh`

    b. From a new terminal window in the same machine, test the endpoint using the command below:
    c. `curl localhost:5000/get-extractors-info?file_type=image/png`

    d. If everything looks fine, kill the Flask app by pressing Ctrl + C.
8. Setup the Flask app as a systemd service

    a. `sudo cp extractor-info-fetcher.service /lib/systemd/system`

    b. `sudo systemctl daemon-reload`

    c. `sudo systemctl start extractor-info-fetcher.service`

9. Check the systemd service status and make sure it is running:

    a. `sudo systemctl status extractor-info-fetcher.service`

10. Update system to allow input traffic to port 5000 in the machine within NCSA network

    a. If the machine is a Nebula OpenStack VM, update iptables by following the steps below:

        i. Create a file in /etc/iptables/rules.d, say *51local.rules* with content:

        ```
        # Extractor Info Service (Flask App)
        -A INPUT -s 141.142.0.0/16 -p tcp -m state --state NEW -m tcp --dport 5000 -j ACCEPT
        ```

        ii. `sudo /etc/iptables/rules.sh`

    b. If the machine is a Nebula OpenStack VM, you will need to create a security group in Nebula based on *51local.rules* mentioned above and add that group to the VM.
11. Try to access http://bd-clowder.ncsa.illinois.edu:5000 or http://bd-clowder-dev.ncsa.illinois.edu:5000 from NCSA network and you should see a welcome message.
12. Now, if you visit http://bd-clowder.ncsa.illinois.edu:5000/get-extractors-info?file_type=image/png or http://bd-clowder-dev.ncsa.illinois.edu:5000/get-extractors-info?file_type=image/png from NCSA network, you should be able to see a JSON formatted array of details about currently running extractors that can process PNG format image files.