|  |  |
| --- | --- |
| **gemini_logo** | **Scratch Install fitsstore** with mod\_wsgi (Apache) |
| **Kenneth Anderson** Paul Hirst |
| Science User Support Department |
| v1.1 – 30 May 2018 |

**Revision History**

v1.0 – 1 March 2018 Kenneth Anderson

v1.1 – 30 May 2018 Kenneth Anderson

**Document ID: GOA-PROC-101\_NewInstallFitsstoreApache**

**Document Purpose**

Instructions for a scratch installation of fitsstore with mod\_wsgi on Apache web server.

**Intended Audience**

Science User Support Department

**Table of Contents**

1. Front Matter 1

2. Initialize Server and ‘fitsdata’ account 2

3. Setup the web server 3

4. Set up and Start Postgresql Database Server 5

5. Detailed Revision History 9

# Front Matter

This is a prescription to install fitsstore under Apache running with mod\_wsgi. mod\_wsgi is the replacement for Apache mod\_python. You will need a user account on a machine to be designated as a fits server. For example, at Gemini-South, we do development on a development server, sbffits-dev-lv1.

Your user account will need `sudo all` authorities. ITS will need to set up this account for you.

You will also need to be set up to use the **fitsdata** account on these servers. The **fitsdata** account will have very limited privileges -- this is the account Apache uses in the run environment.

I would recommend having two windows open, one logged in as **fitsdata** and one as

your own account (w/ sudo). This way, it will be easy to switch from one to the other as

the instructions will specify. (Some things will need sudo authorities, others will not).

These instruction pertain to a "clean install" on a CentOS7 machine; a number of

packages need to be installed (system install) up front. A good amount of the following instruction

set could potentially be placed into a shell script and executed in bulk. For example, all sudo pip install directives could be scripted.

# Initialize Server and ‘fitsdata’ account

Install and set up fitsdata account

Log in to *your* account (with sudo privileges) and begin on a command line:

$ sudo yum -y install httpd httpd-devel mod\_ssl openssl

$ sudo yum -y install httpd postgresql postgresql-server postgresql-devel

On AWS CENTOS7

Install EPEL repository (included by default on ITS config)

$ sudo yum install -y epel-release

$ sudo yum -y install gcc gcc-c++ python-pip python-devel gcc-gfortran cfitsio-devel

Python package installation with pip

$ sudo pip install --upgrade pip

$ sudo pip install mod\_wsgi==4.6.5

$ sudo pip install psycopg2

$ sudo pip install sqlalchemy

$ sudo pip install pyyaml

$ sudo pip install jinja2

$ sudo pip install pyfits

*# pywcs not needed. Now in astropy.*

*We do,*

from astropy import wcs as pywcs

~~$ sudo pip install pywcs~~

$ sudo pip install dateutils

$ sudo pip install requests

$ sudo pip install matplotlib

$ sudo pip install scipy

$ sudo pip install pandas

$ sudo pip install astropy

If using s3 (for dev purposes, probably not)

$ sudo pip install boto3

Set up the 'fitsdata' account

$ sudo /usr/sbin/groupadd -g 5179 fitsdata

$ sudo /usr/sbin/useradd -c 'FITS data' -u 5179 -g 5179 fitsdata

$ sudo passwd fitsdata

If this is a gemini fits server, edit /etc/group and add fitsdata

$ sudo vi(or emacs) /etc/group

geminidata:x:502:fitsdata

Set up FitsStorage and DRAGONS

$ sudo mkdir /opt/FitsStorage

$ sudo chown fitsdata:fitsdata /opt/FitsStorage/

$ cd /opt/FitsStorage

$ svn checkout http://scisoft/svn/FitsStorage/trunk .

Note: If you forget to do this as 'fitsdata', and svn checkout with your sudo account

(it happens!), you can chown the directory recursively.

$ sudo chown -R fitsdata:fitsdata FitsStorage/

Installing DRAGONS

Get DRAGONS from GitHub.

Note: git pull will place 'DRAGONS' directly on cwd

$ cd /opt

$ sudo git clone https://github.com/GeminiDRSoftware/DRAGONS.git

Now, under /opt you should have directories with owners that look like,

drwxr-xr-x 11 fitsdata 4096 Feb 8 16:24 FitsStorage/

drwxr-xr-x 13 <youract> 4096 Feb 7 17:45 DRAGONS

Install fitsverify

As part of file validation, FitsStorage makes use of the executable, "fitsverify'.

This is an executable built from the cfitsio library. You can find this executable

on current production Fits servers in /opt. Add this to /opt on the new server and

make the owner 'fitsdata'. Your /opt directory should then look something like,

/opt:

drwxr-xr-x 12 <youract> root 4096 May 11 16:22 DRAGONS

drwxr-xr-x 12 fitsdata fitsdata 4096 May 8 15:58 FitsStorage

drwxr-xr-x 2 <youract> root 4096 May 20 15:26 fitsverify

It likely will be true that the 'fitsverify' binary will need to be compiled

from source if the OS is something other than CentOS7.

# Setup the web server

Your sudo account will be used to set up the web server.

The following is a shell "install script" that will provide necessary configuration for the Apache server using mod\_wsgi. This is installed with mod\_wsgi-express. You can copy and paste this into a file (suggestion: > /tmp/install\_script.sh) and run it.

(Modify the variables at the beginning of the following script if needed)

# NOTE!!! IF YOU MODIFY THE FOLLOWING, THE CHANGES NEED TO BE

# TRANSLATED TO /etc/systemd/system/fits-httpd.service!!

FITSSTORAGE\_ROOT=/opt/FitsStorage

DRAGONS\_ROOT=/opt/DRAGONS

MODWSGI\_DIR=/opt/modwsgi-default

# Explanation of options that we are setting in MODWSGI\_OPTS

# --server-root Where to store server setup information

# --access-log Enable access log (off by default)

# --url-alias Alias URL for static files (one per alias)

# --python-path Additional content for PYTHONPATH

MODWSGI\_OPTS="\

--server-root $MODWSGI\_DIR \

--port 80 \

--user apache --group apache \

--access-log \

--url-alias /static $FITSSTORAGE\_ROOT/htmldocroot \

--url-alias /favicon.ico $FITSSTORAGE\_ROOT/htmldocroot/favicon.ico \

--url-alias /robots.txt $FITSSTORAGE\_ROOT/htmldocroot/robots.txt \

--python-path $FITSSTORAGE\_ROOT \

--python-path $DRAGONS\_ROOT \

"

mkdir ${MODWSGI\_DIR}

# The extra argument is the WSGI entry point

mod\_wsgi-express setup-server\

$MODWSGI\_OPTS $FITSSTORAGE\_ROOT/fits\_storage/wsgihandler.py

# And set let apache handle that dir from now on

chown -R apache:apache $MODWSGI\_DIR

EOF

**Run the install script to set up mod\_wsgi**

After saving the above install script (here we have written it to /tmp), execute the script.

$ sudo bash /tmp/install\_script.sh

(If you have written the file somewhere else, execute that, of course.)

To run the web server from systemd

$ sudo cp /opt/FitsStorage/otherfiles/etc\_systemd\_system\_fits-httpd.service /etc/systemd/system/fits-httpd.service

$ sudo cp /opt/FitsStorage/otherfiles/etc-sysconfig-httpd /etc/sysconfig/httpd

Edit the config file generated by the install\_script.sh. This configuration file will be

/opt/modwsgi-default/httpd.conf.

Add these lines by the end of the "Alias" list. Then edit to File \* for permissions:

AliasMatch '/(.\*\.html)' '/opt/FitsStorage/htmldocroot/$1'

AliasMatch '/(.\*\.css)' '/opt/FitsStorage/htmldocroot/$1'

AliasMatch '/(.\*\.js)' '/opt/FitsStorage/htmldocroot/$1'

Which should have httpd.conf looking like:

[ … ]

Alias '/favicon.ico' '/opt/FitsStorage/htmldocroot/favicon.ico'

<Directory '/opt/FitsStorage/htmldocroot'>

<Files '\*'>

Require all granted

</Files>

</Directory>

AliasMatch '/(.\*\.html)' '/opt/FitsStorage/htmldocroot/$1'

AliasMatch '/(.\*\.css)' '/opt/FitsStorage/htmldocroot/$1'

AliasMatch '/(.\*\.js)' '/opt/FitsStorage/htmldocroot/$1'

<IfDefine MOD\_WSGI\_VERIFY\_CLIENT>

<Location '/'>

SSLVerifyClient require

SSLVerifyDepth 1

</Location>

</IfDefine>

[ ... ]

In the httpd.conf file, find LimitRequestBody and comment it out:

#LimitRequestBody 200000000

Comment out all the \_default\_ VirtualHost definitions.

E.g.,

<IfDefine !MOD\_WSGI\_VIRTUAL\_HOST>

<IfVersion < 2.4>

NameVirtualHost \*:80

</IfVersion>

**#<VirtualHost \_default\_:80>**

**#</VirtualHost>**

</IfDefine>

Be sure to comment a full element block – do not create badly formed xml. The next

virtual host element should look like,

<IfDefine MOD\_WSGI\_VIRTUAL\_HOST>

<IfVersion < 2.4>

NameVirtualHost \*:80

</IfVersion>

**#<VirtualHost \_default\_:80>**

**#<Location />**

**#Order deny,allow**

**#Deny from all**

**#<IfDefine MOD\_WSGI\_ALLOW\_LOCALHOST>**

**#Allow from localhost**

**#</IfDefine>**

**#</Location>**

**#</VirtualHost>**

# 

# Edit /opt/modwsgi-default/handler.wsgi

Set "entry point". The default install of modwsgi-default sets the entry point in the handler.wsgi file to be

entry\_point = '/opt/modwsgi-default/default.wsgi'

Set this to use the FitsStorage wsgihandler.py:

entry\_point = '/opt/FitsStorage/fits\_storage/wsgihandler.py'

# 4. Set up and Start Postgresql Database Server

**Set up log and backup directories**

$ sudo mkdir /data/logs

$ sudo chown fitsdata /data/logs

$ sudo mkdir /data/backups

$ sudo chown fitsdata /data/backups

$ sudo mkdir /data/upload\_staging

$ sudo chown fitsdata /data/upload\_staging

$ sudo chmod oug+rwx /data/upload\_staging

$ sudo mkdir /data/z\_staging

$ sudo chown fitsdata /data/z\_staging

If using s3:

$ sudo mkdir /data/s3\_staging

$ sudo chown fitsdata /data/s3\_staging

Apache needs access to this to download preview images now

$ sudo chmod oug+rw /data/s3\_staging

Set up the postgresql data directory

$ sudo mkdir /data/pgsql\_data

$ sudo chown postgres:postgres /data/pgsql\_data

$ sudo cp /lib/systemd/system/postgresql.service /etc/systemd/system/

Edit the postgresql.service file just placed in /etc/systemd/system/ and add the PGDATA environment var:

$ sudo vi (emacs) /etc/systemd/system/postgresql.service

Environment=PGDATA=/data/pgsql\_data

(save & exit)

Initiate the database

$ sudo systemctl daemon-reload

$ sudo postgresql-setup initdb

$ sudo systemctl start postgresql.service

$ sudo su - postgres

$ /usr/bin/createuser --no-superuser --no-createrole --createdb fitsdata

$ /usr/bin/createuser --no-superuser --no-createrole --no-createdb apache

Edit the postgresql configuration and set the effective cache size

$ emacs (vi) /data/pgsql\_data/postgresql.conf

#shared\_buffers = 256MB

# 25% of memory if memory > 1GB

effective\_cache\_size = 8000MB

Save and exit.

If there will be remote access to this database (eg for eavedropping), edit postgresql.conf and set,

listen\_addresses = '\*'

And edit /data/pgsql\_data/pg\_hba.conf . At end of file, add a comment line to say what you're adding. E.g.,

host fitsdata all 10.1.5.47/32 trust exit

Save the file and exit.

**Start Postgresql**

$ sudo systemctl restart postgresql.service

Need to set postgresql and httpd to start at boot

$ sudo systemctl enable postgresql.service

$ sudo systemctl enable fits-httpd.service

Now get a shell logged in as ‘fitsdata’

$ cd /opt/FitsStorage/fits\_storage

$ emacs (vi) fits\_storage\_config.py

# Set values

# This is a setup for a fitssstore server, not the archive. Set that boolean:

use\_as\_archive = False

# Set the storage root to point to a data source. For testing, place test

# data under /data/

storage\_root = '/data/gemini\_data'

# On production servers, this will point to dataflow:

storage\_root = '/sci/dataflow'

(save and exit)

Create db and set up tables.

$ /usr/bin/createdb fitsdata

$ python scripts/create\_tables.py

Should now be good to go.

* Check with ITS if you need a whitelisted local mail relay on it
* Check with ITS that the development/operations status of the machine

To run jobs from system

$ sudo cp /opt/FitsStorage/otherfiles/etc\_systemd\_system\_fits-service\_ingest\_queue1.service /etc/systemd/system/fits-service\_ingest\_queue1.service

$ sudo systemctl start fits-service\_ingest\_queue1

$ sudo systemctl enable fits-service\_ingest\_queue1

Copy again and edit for siq2 if you want two of them

-- Same for 2 if you want it

-- Repeat for export queue and api

# 5. Detailed Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| v1.0 | 1 March 2018 | | Kenneth Anderson |
| v1.1 | 30 May 2018 | | Kenneth Anderson |
|  |  | |
|  |  | |  |
|  |  | |
|  |  | |  |
|  |  | |