**Deploying the Rapid Website to a New Instance**

**Installation and Configuring the Web server:**

1. Log into the AWS Ubuntu Web Server
2. Checkout the RAPID repo from github

$git clone <https://github.com/gdit-cnd/RAPID> RAPID

1. Checkout the develop or desired branch

Git checkout <branch\_name>

1. Create a secrets.json file in the $HOME\RAPID directory and copy and paste the contents from the production version.
2. Update the correct path for secrets.json in $HOME\RAPID\settings\base.py file

Ie., with open(“**/<home>/ RAPID/secrets.json**”, ”r”)

1. Update the $HOME\RAPID\settings\local.py file with the correct environment info

BASE\_SITE\_URL = ‘http://0.0.0.0:8000”sudo

ALLOWED\_HOSTS = [’0.0.0.0’]

LOGGING = {

…

'file': {

…

'filename': '/<home>/ RAPID/RAPID.log',

}…

1. Update the $HOME\RAPID\settings\production.py file

BASE\_SITE\_URL = ‘http://0.0.0.0”

ALLOWED\_HOSTS = [’0.0.0.0’, ‘ec2-0-0-0-0.us-west-2.compute.amazonaws.com']

ADMINS = (('RAPID-Admin’, 'rapidpivot@gmail.com'),)

LOGGING = {

…

'file': {

…

'filename':'/<home>/RAPID/RAPID.log',

1. Update the APPLICATION\_DIR parameter in the install.sh file
2. Add permissions to $HOME\RAPID\install.sh

* **chmod +x install.sh**

1. Run install.sh as administrator

* **sudo ./install.sh**

1. Create symbolic links for libraries.
   1. Create the folder structure /var/www/rapid
   2. In the folder /var/www/rapid, create a symbolic link for the media and static physical folder structures using the sudo ln -s command

EXAMPLE:

sudo ln -s /<home>/RAPID/static/DataTables-1.10.5/media /var/www/rapid/media

sudo ln -s /<home>/RAPID/static /var/www/rapid/static

1. Generate and Configure SSL Certs

Copy the production SSL certs over to the following locations /etc/apache2/ssl

As an alternative, you can generate SSL Certs (self-signed certificate) using openssl

(Please reference link https://www.digitalocean.com/community/tutorials/how-to-create-a-ssl-certificate-on-apache-for-ubuntu-14-04)

1. Create the /etc/apache2/ssl directory

sudo mkdir /etc/apache2/ssl

1. Change directory to /etc/apache2/ssl and generate key and certificate using openssl

cd /etc/apache2/ssl

EXAMPLE:

sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/apache2/ssl/apache.key -out /etc/apache2/ssl/apache.crt

When you hit enter, you will be asked a number of questions.

Country Name (2 letter code) [AU]:US

State or Province Name (full name) [Some-State]:Virginia

Locality Name (eg, city) []:Reston

Organization Name (eg, company) [Internet Widgits Pty Ltd]:DIA

Organizational Unit Name (eg, section) []:CND

Common Name (e.g. server FQDN or YOUR name) []:<Webserver\_IP\_Address>

Email Address []:

verify following files are created in the /etc/apache2/ssl directory

apache.key

apache.crt

1. Configure Rapid configuration files
   1. Copy the following example templates located at: /external\_configs/apache2/ into the /etc/apache2/sites-available directory

rapid-ssl.conf

rapid.conf

* 1. Update the rapid.conf and rapid-ssl.conf templates located in the /etc/apache2/sites-available directory

WSGIDaemonProcess RAPID python-path=/<home>/RAPID

WSGIScriptAlias / /<home>/RAPID/RAPID/wsgi.py

RedirectPermanent / https://ec2-0-0-0-0.us-west-2.compute.amazonaws.com

ServerAdmin rapidpivot@gmail.com

ServerName ec2-0-0-0-0.us-west-2.compute.amazonaws.com

ServerAlias www.RapidDev.com

<Directory /<home>/RAPID/RAPID>

<Files wsgi.py>

Require all granted

</Files>

</Directory>

* 1. Enable the SSL module to active it

sudo a2enmod ssl

* 1. Activate the SSL Virtual Host to use custom templates

sudo a2ensite rapid-ssl.conf

sudo a2ensite rapid.conf

To make future updates to the configuration files, you need to disable and then renable the SSL Virtual Host to use the templates. Use the following commands to disable the templates.

sudo a2dissite rapid-ssl.conf

sudo a2dissite rapid.conf

* 1. Restart Apache to load the new virtual host files

sudo /etc/init.d apache2 reload

sudo /etc/init.d/apache2 restart

1. Check that the celery RabbitMQ server is running

>sudo rabbitmqctl status

If it’s not, then start the RabbitMQ server

>sudo /etc/init.d/rabbitmq-server start

>sudo systemctl start rabbitmq-server

NOTE: If the Rabbit service is corrupted, deinstall and reinstall the program

>sudo apt-get remove rabbitmq-server

>sudo apt-get install rabbitmq-server

NOTE: If Celery service is corrupted, run /RAPID/install.sh again to reinstall celery and dependencies

You may get a message that says

Warning: celery\_pivoteer.service changed on disk. Run ‘systemctl daemon-reload’ to reload units

>systemctl daemon-reload

1. Configure Celery configuration files

Change the permissions on the following files (ie., sudo chmod 640)

/etc/default/celery\_beat

/etc/default/celery\_daemon

/etc/default/celery\_pivoteer

1. Edit the CHDIR parameter in the following files to point to the working space directory -CHDIR (/<home>/RAPID)

/etc/default/celery\_beat

/etc/default/celery\_daemon

/etc/default/celery\_pivoteer

1. Start the celery worker processes

> sudo service celery\_beat start

> sudo service celery\_daemon start

> sudo service celery\_pivoteer start

> sudo etc/init.d/celery\_beat start

> sudo etc/init.d/celery\_daemon start

> sudo etc/init.d/celery\_pivoteer start

1. Copy the GeoLite2-City.mmdb from the production server- /apps/RAPID/core folder to the new server /<home>/RAPID/core folder
2. Add a rule to the associated Security Group for the AWS server

Type-SSH Protocol -TCP Port-22 Source- 0.0.0.0/0

Type-HTTPS Protocol-TCP Port-443 Source-0.0.0.0/0

1. Add a rule to the associated Security Group for the AWS Database RDS server

Type-PostgreSQL Protocol-TCP Port-5432 Source-<web server’s IP>

**Installation and Configuring the Database server:**

1. Log into the AWS production server and create a backup of the database

* **sudo su – postgres**
* **Pg\_dump rapid > rapid\_prod.dump (rapid\_prod\_18feb.dump)**

Store the backup file - rapid\_prod.dump off to an accessible location to be referenced later.

You can also use winscp to download/upload the backup file. The destination folder needs to be pointing to the $Home folder (/home/ubuntu). The transfer mode needs to be in Binary with Ignore permission errors checked.

1. If logging into a local instance of the PostgreSQL db, create a new PostgreSQL superuser account and set password (should be assigned to SQL\_USER and SQL\_PASS respectively in the secrets.json file):

* **sudo su - postgres**
* **createuser rapid - -pwprompt**
* **psql**
* **ALTER USER rapid CREATEDB**

1. If logging into a local PostgreSQL db, use the follow command to log in as the postgres user

* **psql**

Note: To log in as the rapid user ….

* **psql --host=localhost ---username=rapid –dbname=<database\_name>**
* **Enter the password for the username:**

1. If logging into an external AWS RDS server, use the follow command to log in as the postgres user

* **psql --host=database.server.rds.amazonaws.com --port=5432 --username postgres**

Note: To log in as the rapid user ….

* **psql --host= database.server.rds.amazonaws.com --port=5432 –username=rapid -dbname=postgres**
* **Enter the password for the username:**

Note: To test the connection to the external AWS RDS server

* **nc -zq database.server.rds.amazonaws.com 5432**

1. Create the database rapid as the postgres user

* **CREATE DATABASE rapid**

Confirm the database is created by typing the followipng at the command prompt

* **\l**

A listing of the databases will be displayed as follows…

List of databases

Name | Owner | Encoding | Collate | Ctype | Access privileges

-----------+----------+----------+-------------+-------------+-----------------------

postgres | rapid | UTF8 | en\_US.UTF-8 | en\_US.UTF-8 |

rapid | rapid | UTF8 | en\_US.UTF-8 | en\_US.UTF-8 |

1. Connect to the rapid database as the owner

* **\connect rapid**

1. Restore the production database backup to the rapid database

* **\i /home/Ubuntu/data/rapid\_prod.dump**

The screen will display with a list of DML commands and counts. It will take about 5 minutes.\q

1. Verify the data has been uploaded by typing the following at the command prompt

* **\dt**

A listing of the database relations will be displayed as follows… There should be 22 rows in all

List of relations

Schema | Name | Type | Owner | Size | Descri:qption

--------+----------------------------------+-------+-------+------------+-------------

public | auth\_group | table | rapid | 0 bytes |

1. You can also query on the data to confirm that the tables are populated. The counts should match those of production.

* **Select count(\*) from monitors\_certificatemonitor;**

**Count should be >=110**

* **Select count(\*) from profiles\_profile;**

**Count should be >= 361 records**

1. To exit the out of the psql prompt. This will close to the connection to the database prompt and return to the webserver

* **\q**

1. Open a new SSH or terminal session and connect to the AWS web server.
2. Navigate to the $HOME/Rapid folder.
3. Update the secrets.json file with the correct database connection string and authentication info:

"SQL\_NAME": "rapid",

"SQL\_HOST": "rapid-t2large.cjuelvk2x7js.us-west-2.rds.amazonaws.com",

"SQL\_USER": "rapid",

"SQL\_PASS": "rapidrapid",

1. Verify the following files are in the \apps\RAPID\monitors\migrations folder

**0003\_create\_subscriptions.py**

**0004\_migrate\_subscriptions.py**

1. Synchronize the database with the code updates:

* **sudo python3 manage.py syncdb**

**Confirm the following messages are displayed:**

Applying monitors.0003\_create\_subscriptions

Applying monitors.0004\_migrate\_subscriptions

Populating certificate subscriptions

Populating domain subscriptions

Populating IP subscription

OK

**If you are getting error messages, then you need to generate the migration scripts and migrate them (Follow steps 16-18). If the database is correctly updated, then skip to step 19.**

1. Create migrations for the new code:

* **sudo python3 manage.py makemigrations**

1. Confirm the new migration files are created and also created in <home>/RAPID/monitors/migrations folder

* **sudo python3 manage.py showmigrations**

1. Execute the new migration files:

* **sudo python3 manage.py migrate**

1. Confirm the database has been updated by returning to the database server and entering the following command at the postgres psql prompt

* **\dt**

A listing of the database relations will be displayed as follows. Confirm the following new tables are created

List of relations

Schema | Name | Type | Owner | Size | Description

--------+----------------------------------+-------+-------+------------+-------------

public | monitors\_certificatesubscription| table | rapid | 0 bytes |

public | monitors\_domainsubscription| table | rapid | 0 bytes |

public | monitors\_ipsubscription| table | rapid | 0 bytes |

Run the following query at the psql prompt to confirm the tables are populated

* **Select count(\*) from monitors\_certificatesubscription;**
* **Select count(\*) from monitors\_domainsubscription;**
* **Select count(\*) from monitors\_ipsubscription;**

The rowcount should be greater or equal to the count from the respective tables beow:

* **Select count(\*) from monitors\_certificatemonitor;**
* **Select count(\*) from monitors\_domainmonitor;**
* **Select count(\*) from monitors\_ipmonitor;**
* **Select \* from Django\_migrations**

**Confirm that the following migrations have been added listed:**

**0003\_create\_subscriptions**

**0004\_migration\_subscriptions**

1. Run the following query to check for bad data in the tag table

* **select \* from monitors\_certificatemonitor\_tags**

**where certificatemonitor\_id not in**

**(select certificate\_id from monitors\_certificatesubscription) order by certificatemonitor\_id;**

Clean up any bad certificate values in the tag table

* **update monitors\_certificatemonitor\_tags**

**set certificatemonitor\_id = replace(replace(certificatemonitor\_id,'',''),' ','');**

1. Run the following query to check for bad data that did not get migrated over

* select certificate\_value from monitors\_certificatemonitor

where certificate\_value not in

(select certificate\_id from monitors\_certificatesubscription)

And certificate\_value not in

(select certificatemonitor\_id from monitors\_certificatemonitor\_tags)

order by certificate\_value;

Execute the following query to delete any bad data that did not get migrated over

* delete from monitors\_certificatemonitor

where certificate\_value not in

(select certificate\_id from monitors\_certificatesubscription)

And certificate\_value not in

(select certificatemonitor\_id from monitors\_certificatemonitor\_tags)

1. Restart the apache and celery services on the webserver. This needs to be done on the webserver in the following order:

(stopall.sh)

* **sudo /etc/init.d/celery\_beat stop**
* **sudo /etc/init.d/celery\_daemon stop**
* **sudo /etc/init.d/celery\_pivoteer stop**
* **sudo /etc/init.d/apache2 stop**

(startall.sh)

* **sudo /etc/init.d/apache2 start**
* **sudo /etc/init.d/celery\_pivoteer restart**
* **sudo /etc/init.d/celery\_daemon restart**
* **sudo /etc/init.d/celery\_beat restart**

**Installing Miscellaneous Libraries:**

**Geckodriver v0.16.1**

1. At the command prompt on the web server, type in the following

* **Wget https://github.com/mozilla/geckodriver/releases/download/v0.16.1/geckodriver-v0.16.1-linux64.tar.gz**
* **cp geckodriver-v0.16.1-linux64.tar.gz /usr/local/lib/python3.5/dist-packages/**
* **cd to /usr/local/lib/python3.5/dist-packages/**
* **tar -xvzf geckodriver-v0.16.1-linux64.tar.gz**
* **confirm geckodriver is created in /usr/local/lib/python3.5/dist-packages**

**Selenium v3.4.2**

1. At the command prompt on the web server, type in the following

* **sudo apt-get install selenium**
* **confirm selenium-3.4.2.dist-info is created in /usr/local/lib/python3.5/dist-packages**

**Firefox v53.0.2**