**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”

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

* **sudo ./install.sh**

1. Generate and Configure SSL Certs

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. Enable the SSL module to activate it

sudo a2enmod ssl

1. Restart the web server for the change to be recognized

sudo service apache2 restart

sudo /etc/init.d/apache2 restart

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) []:dodiis.mil

Email Address []:

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

apache.key

apache.crt

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. 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

Cd /

<Directory /<home>/RAPID/RAPID>

<Files wsgi.py>

Require all granted

</Files>

</Directory>

* 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 service apache2 reload

sudo service apache2 restart

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 /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)**

1. From the production server, test the connection to the new database server

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

1. From the production server, log into the AWS RDS Web Server as the postgres user

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

Or you can log directly in as the rapid user ….

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

1. Create the database rapid as postgres user

* **CREATE DATABASE rapid**

Confirm the database is created by typing the following 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

* **\connect rapid**

1. Restore the production database backup to the rapid database

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

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 | Description

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

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 >=107**

* **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": "devtest.cjuelvk2x7js.us-west-2.rds.amazonaws.com",

"SQL\_USER": "rapid",

"SQL\_PASS": "rapidrapid",

1. Synchronize the database with the code updates:

* **sudo python3 manage.py syncdb**

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 filder

* **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 to confirm the table monitors\_certificatesubscription is empty

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

1. Copy the file 000x\_migrate\_subscriptions.py from $HOME\RAPID\monitors\migrations\post to $HOME\RAPID\monitors\migrations directory

* **cp 000x\_migrate\_subscriptions.py ../**
* **cp 000x\_migrate\_domain\_ip\_subscriptions.py ../**

1. Edit the file **000x\_migrate\_subscriptions.py** with the name of the last migration file generated

dependencies = [

('monitors', **'0003\_auto\_20170221\_1255.py'**),

]

1. Edit the file **000x\_migrate\_domain\_ip\_subscriptions.py** with the name of the last migration file generated

dependencies = [

('monitors', **'000x\_migrate\_subscriptions.py'**),

]

1. Execute the above migration files:

* **sudo python3 manage.py migrate**

Run the following query at the psql prompt to confirm the table monitors\_certificatesubscription is now populated

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

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

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

1. Restart the apache service on the webserver

* **sudo /etc/init.d/apache2 restart**