**28 JUN 2015**

# SOFTWARE INSTALLATION GUIDE (SIG)
# ADVERSE DRUG EFFECTS RESEARCH SYSTEM (ADERS) PROTOTYPE

**Submitted in Response to:**

## Agile Service Delivery

## 4QTFHS150004

**For:**

**The General Services Administration (GSA)**
**Federal Acquisition Service, Integrated Technology Service**
**National IT Commodity Program**
**401 West Peachtree Street NW, Suite 820**
**Atlanta, GA 30308**

**By:**

**Intellect Solutions LLC**
**GSA Contract # GS-35F-144AA**
**5602 Pickwick Road**
**Centreville, VA 20120**
**Dr. Sunny Singh, COO, Phone: (703) 898-1498**

# DOCUMENT APPROVAL

**SUBMITTED BY:**

_____ _____
{Author Name} DATE
{Author Position Title}
{Organization}

**APPROVED BY:**

_____ _____
{Name} DATE
{Position Title}
{Organization}

_____ _____
{Name} DATE
{Position Title}
{Organization}

**CONCURRED BY:**

_____ _____
{Name} DATE
{Position Title}
{Organization}

# DOCUMENT HISTORY

| Version/Document Number | Description | Author |
|---|---|---|
| Intellect SIG-ADERS Prototype | Initial Submission for Software Installation Guide (SIG) Adverse Drug Effects Research System (ADERS). | 28 June 2015 |

# TABLE OF CONTENTS

# LIST OF FIGURES

**This Page intentionally Left Blank**

# 1.0 INTRODUCTION

## 1.1 PURPOSE

This Software Installation Guide (SIG) contains instructions for installing and configuring the Adverse Drug Effects Research System (ADERS) Prototype componets onto the Amazon Web Service (AWS) infrastructure. These instructions cover the AWS Server Instance creation, Apache and Node JS installation and configuration, and the download, installation, and configuration of the ADERS OpenSource prototype system from GitHub. This SIG contains the following Sections.

- Section 1.0 is a general Introduction to this SIG.
- Section 2.0 contains a listing of the roles and responsibilities of the Installer.
- Section 3.0 contains a system overview for the ADERS system.
- Section 4.0 contains an Overview, list of Assumptions and Prerequisites for installing ADERS
- Section 5.0 contains the detailed instructions for standing up an AWS server for hosting the ADERS system.
- Section 6.0 contains instructions for establishing a secure shell (SSH) connection to the AWS server instance created.
- Section **Error! Reference source not found.** contains the server customization instructions including the docker commands to run the ADERS containers.
- Appendix A contains a list of Acronyms used in this document.

## 1.2 REFERENCES

- Amazon Linux (AMI) Release Notes (https://aws.amazon.com/amazon-linux-ami/2015.03-release-notes/)
- Node JS (https://nodejs.org)
- Apache (https://www.apache.org)
- ADERS GitHub Respository (https://github.com/IntellectSolutions-GSA-Prototype/OpenFDA_Prototype)

## 1.3 SCOPE

The scope of this document is ADERS Prototype initial deployment. This document is intended for all interested stakeholders of the ADERS / OpenFDA Program and provides installation instructions for the ADERS Web Application and support components.

Additional details can be found in the documents listed in Section 1.2.

## 1.4 AUTHENTICATION

Authority to formulate changes to this document prior to delivery is designated to the ADERS Program Manager.

## 1.5 REVIEW

This document is reviewed and revised as needed when the system is modified. Version control of the document is coordinated through Intellect Solutions under Configuration Management (CM) controls and procedures.

## 2.0 ROLES AND RESPONSIBILITIES

**Key Role Assignment for this installation: Site-designated Installer**.

These installation instructions do not assume that the Installer has experience with installing this software; however, they do assume that the Installer possesses some technical expertise and has rights to perform tasks on the system. The assumptions are as follows:

1. The Installer must have at least a novice level of familiarity with:
    a. the Linux Operating System;
    b. Command Line Interfaces (CLI) including Command Line Editors (e.g., vi);
    c. Amazon Web Services Administration
    d. Docker commands for building, maintaining, starting, and stopping images and containers.
2. The Installer has Administrator login credentials and rights to the Amazon Web Services and Linux Server.

**This Page intentionally Left Blank**

# 3.0 SYSTEM OVERVIEW

Figure 3-1, an architectural overview originally developed by Architecture Group, shows the use of ADERS as a bridge for users querying OpenFDA to provide additional privacy protection. The functionality this adds to OpenFDA by: Users select a query to retrieve information related to generic or brand name drugs that are reporting adverse effects to one or more demographic groups. The information is relayed securely to the ADERS server which then passes the properly formated OpenFDA query to the api.fda.gov interface and returns the result to the client.

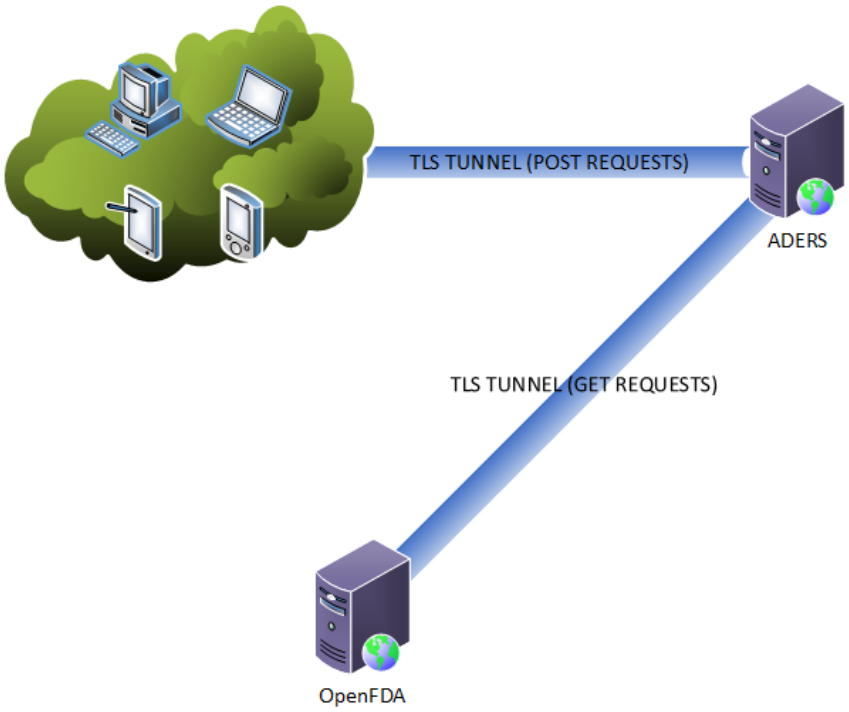



**Figure 3-1: ADERS Architectural Overview**

**This Page intentionally Left Blank**

# 4.0 OVERVIEW, LIST OF ASSUMPTIONS AND PREREQUISITES

## 4.1 OVERVIEW

☑ This Installation Guide provides detailed steps on how to configure the **Amazon Web Service (AWS) Linux Server** for hosting the ADERS web application.

☑ ADERS is comprised of two software components:

1. Apache Web Server
2. Node.JS Server

☑ ADERS repository consists of two primary directories (Front End and Back End) containing the files required for supporting ADERS on the Apache (Front End) and Node.JS (Back End) components.

☑ Installation and configuration takes approximately one hour. ***Please read all instructions before proceeding with the installation***.

## 4.2 ASSUMPTIONS

☑ A privileged account capable of launching and configuring AWS Instances is already available for supporting this effort (the creation of an AWS user account is not convered in this document).

☑ The installer has a working knowledge of installing and configuring packages in a Linux Environment (e.g., rpm, yum, apt-get).

☑ The installer is familiar with line editors and command line interfaces in Linux.

## 4.3 PREREQUISITES

The following software products should already be installed before continuing with the installation:

☑ SSH Client (e.g. putty)

☑ SSH Key Generator (e.g., puttygen)

**This Page intentionally Left Blank**

# 5.0 AWS SERVER INSTANACE LAUNCH

## 5.1 STEP 1 – CHOOSE AN AMAZON MACHINE IMAGE (AMI)

Login to the **AWS Portal** with an Administrator account.

Access the EC2 Service Portal

Select "Launch Instance"

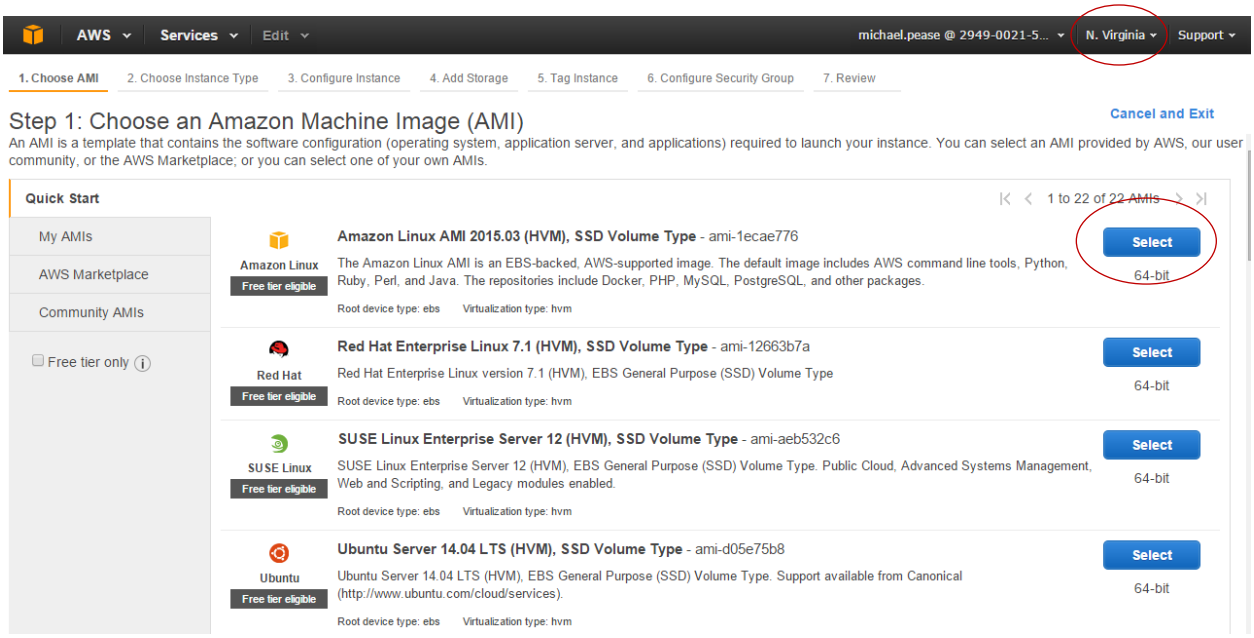



**Figure 5-1: Choosing an Amazon Machine Image (AMI)**

As shown in Figure 5-1: Choosing an Amazon Machine Image (AMI), be certain that the region is set correctly (N. Virgina was selected for this prototype), and then select the Amazon Linux AMI 2015.03 (HVM) (or latest version version available).

## 5.2 STEP 2 – CHOOSE AN INSTANCT TYPE

AWS provide several instance types which define the processing, memory, and storage available for the virtual server.

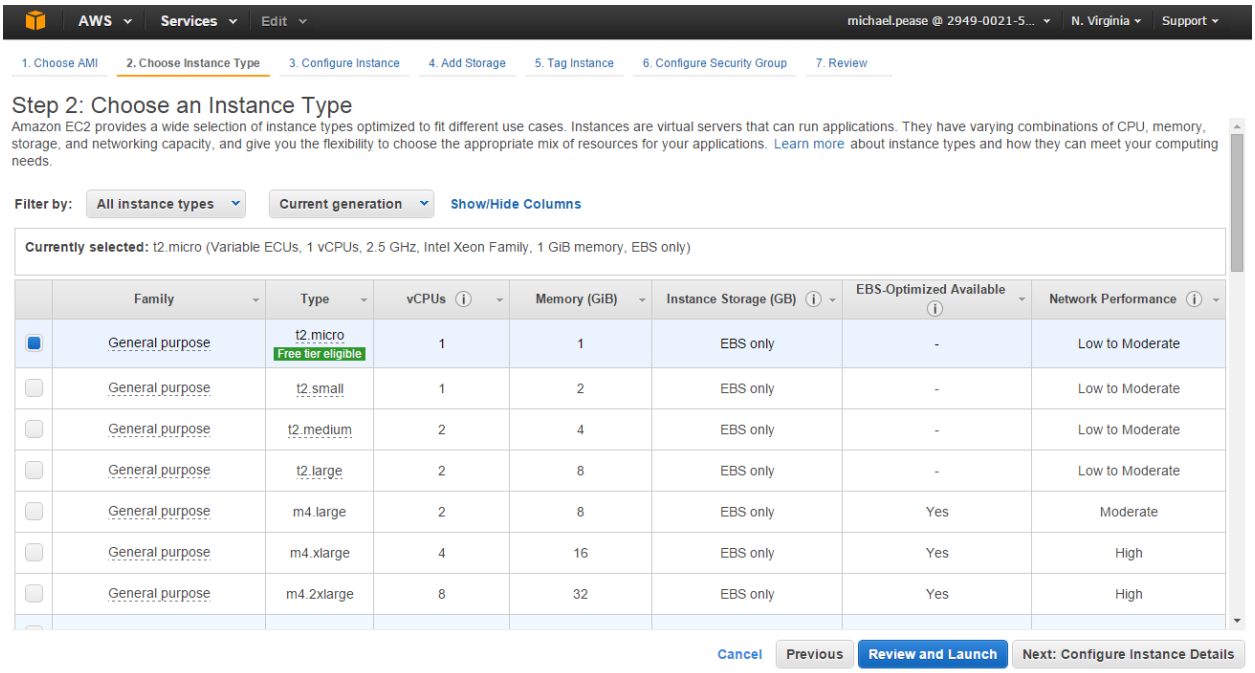



**Figure 5-2: Instance Type Selection**

For the ADERS prototype the General Purpose t2.micro instance type was choosen. Production instances and full-scale test instances would use the type that has been determined to meet the expected user load. At this time, the t2.medium is the server configuration expected for supporting production and full-scale testing.

## 5.3 STEP 3 – CONFIGURE INSTANCE

The next step involves setting instance specific roles and networking options. As shown below, the network and IAM Roles should be selected to ensure proper configuration.

| | |
|---|---|
| **SPECIAL NOTE:** | Make sure Shutdown Behavior is set to "Stop" and "Protect Against Accidental Termination" is enabled. AWS will terminate (delete) instances resulting in potential data loss if these settings are not correct. |

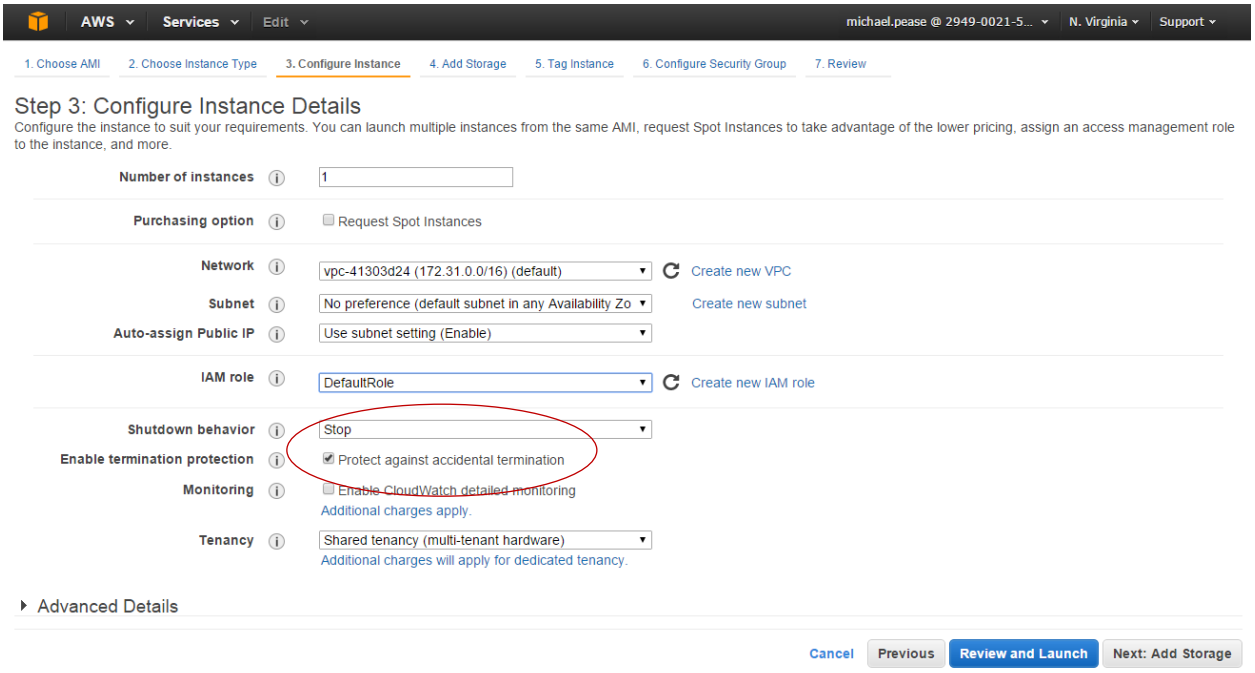



**Figure 5-3: Configure Instance Options**

## 5.4 STEP 4 – ADD STORAGE

The default storage setting (8 GB) is sufficient for the ADERS prototype and for the production instances. Additional storage maybe required if additional logging is enabled.

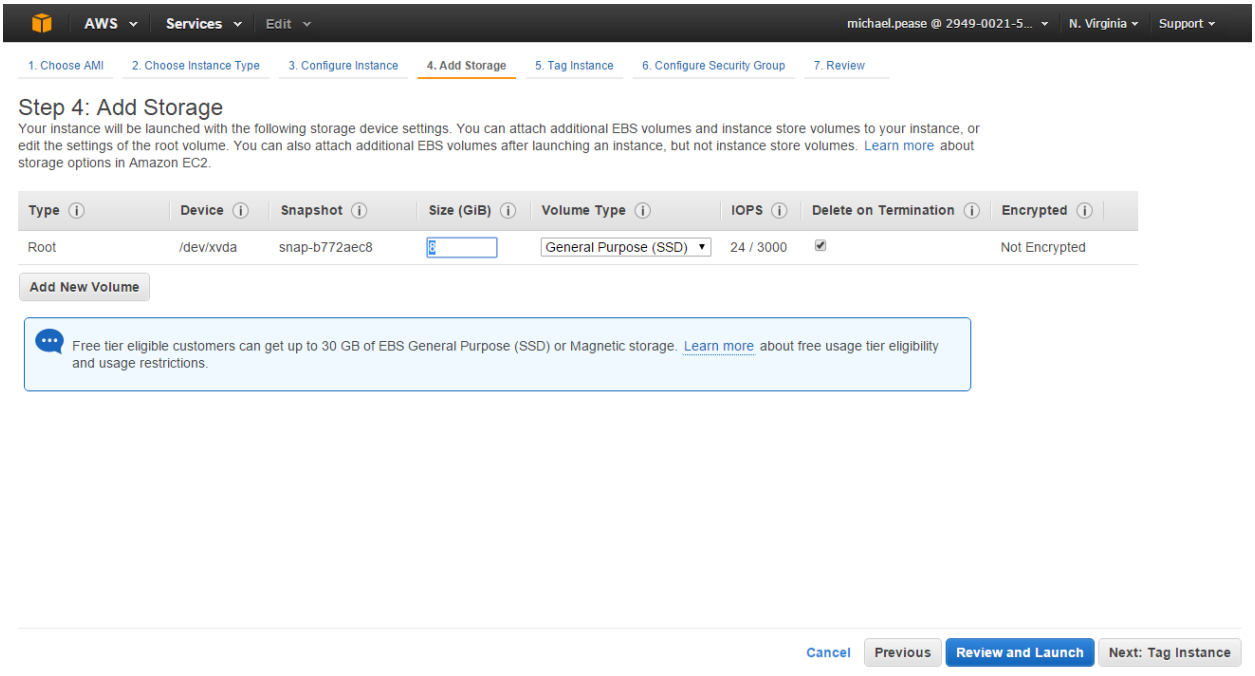



**Figure 5-4: Storage Parameters**

## 5.5 STEP 5 – TAG INSTANCE

Instances can have meta data tags associated with them to make management and monitoring easier in large scale deployment environments. Place an appropriate name for the instance in the available field.

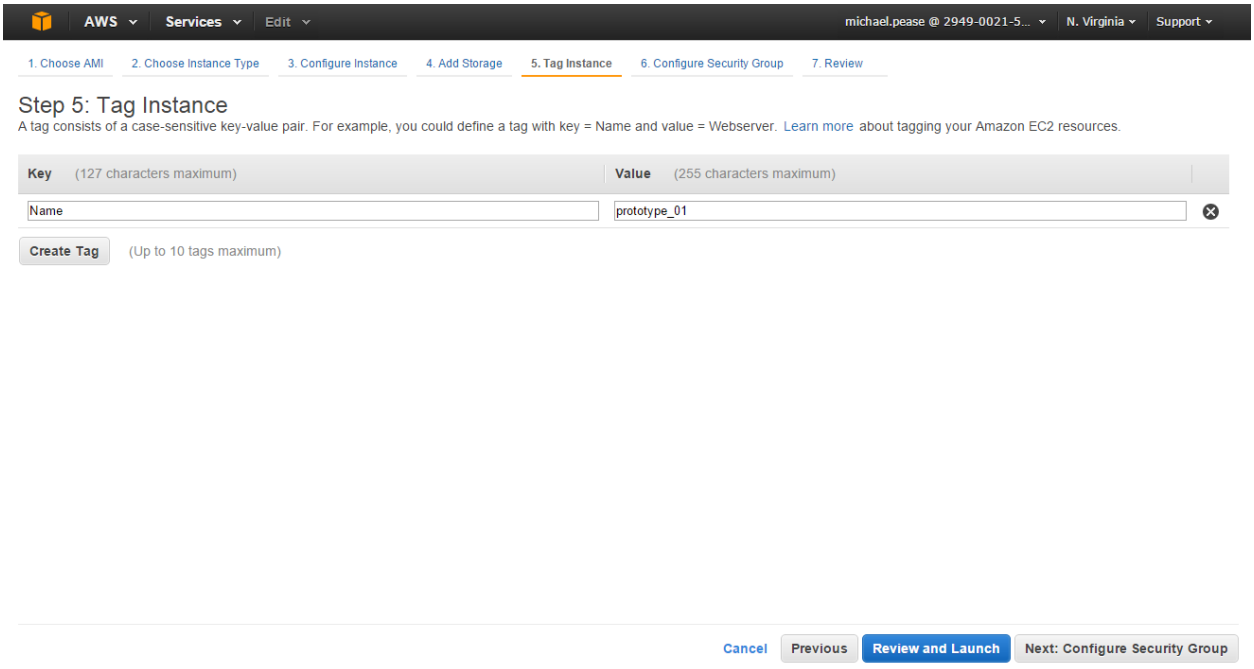



**Figure 5-5: Tag Parameters**

## 5.6 STEP 6 – CONFIGURE SECURITY GROUPS

For the site to function properly the following ports are required:

1. TCP/22 – Secure Shell; for remote access and administration
2. TCP/80 – Standard HTTP Traffic
3. TCP/443 – Standard HTTPS Traffice
4. TCP/8000 – Node JS port configured for SSL Communication Only

| **SPECIAL NOTE:** | For added security, SSH connection should be restricted to known IP Addresses only. |
|---|---|

The resulting rule set is depicted below:

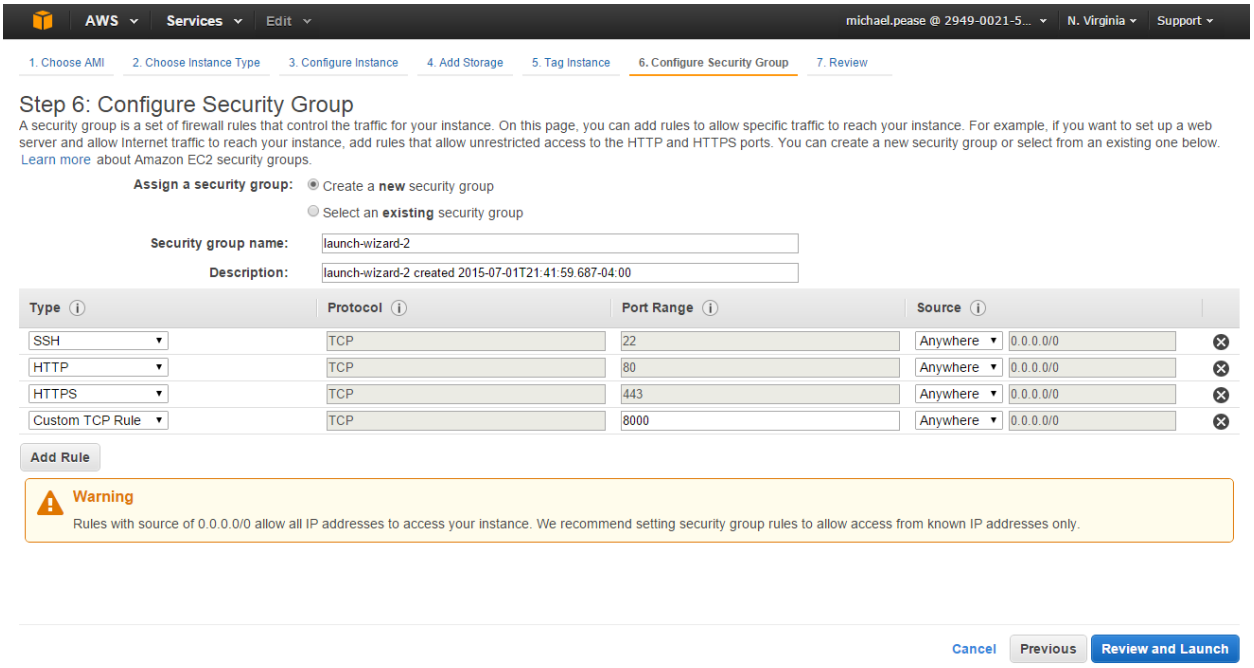



**Figure 5-6: Security Group Configuration**

## 5.7 STEP 7 – REVIEW AND LAUNCH

This is a critical step in the process. Use the available information to validate that the instance settings are correct and then launch the instance.

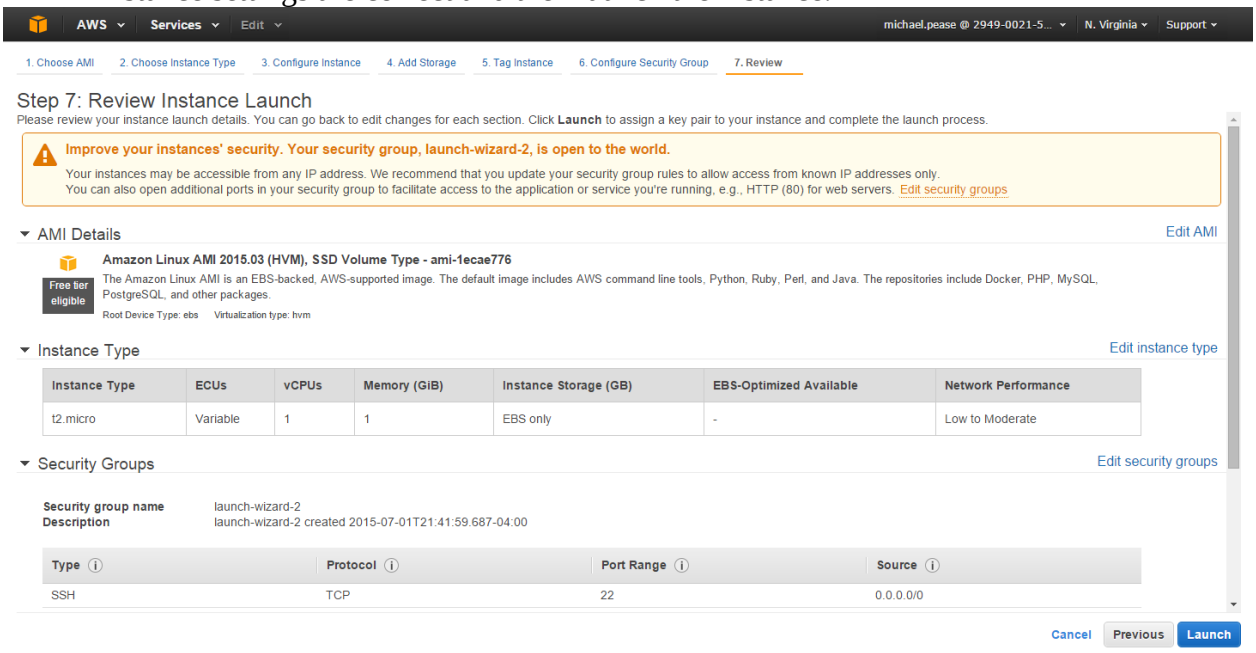



**Figure 5-7: Security Group Configuration**

**SPECIAL NOTE:** As mentioned in Step 6, for production instance, SSH protocol must be restricted to known IP Addresses/Address Ranges.

After launching the instance, the EC2 portal should show the running instance and metadata for the system and monitoring.

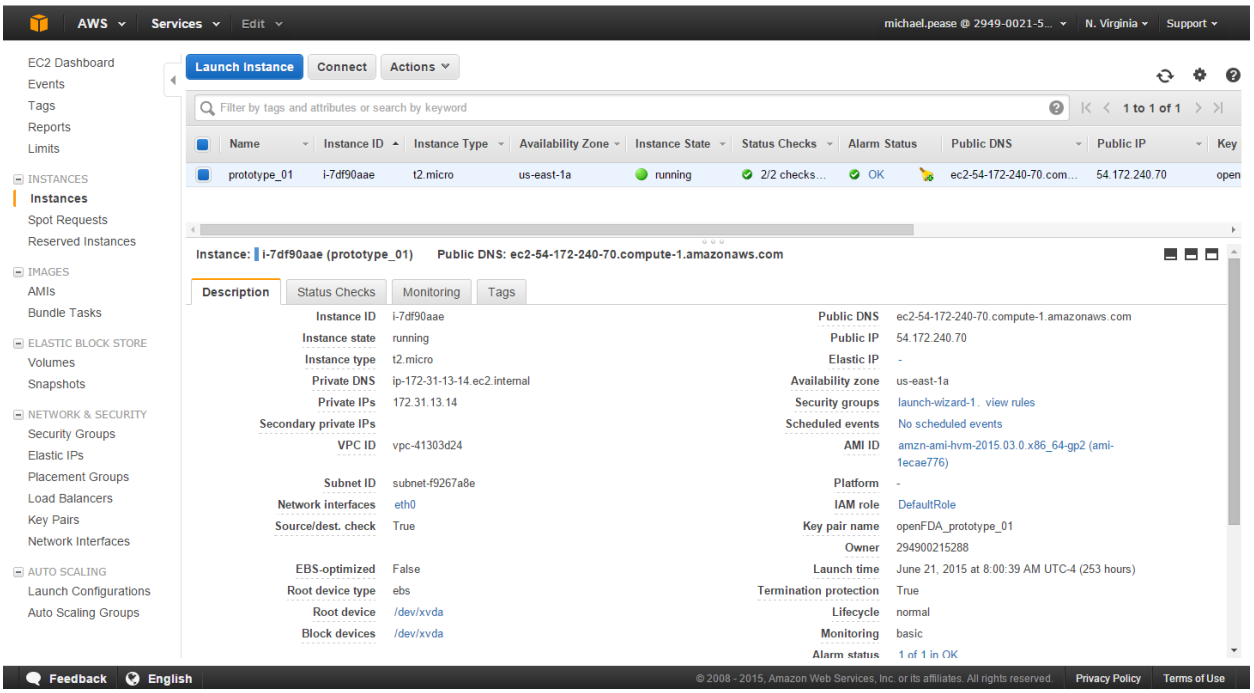
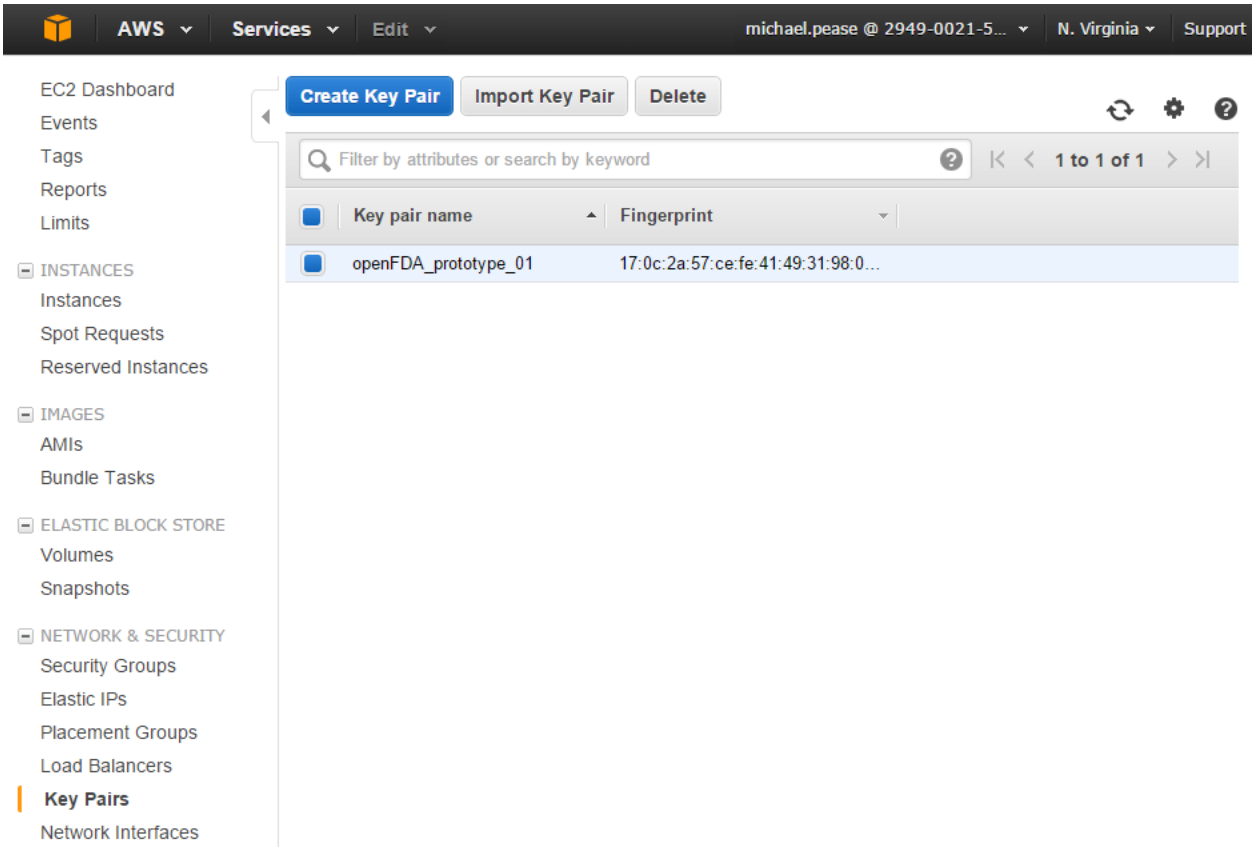



**Figure 5-8: Security Group Configuration**

## 6.0 CONNECTING TO LINUX INSTANCE

When the instance is create, a default privileged account (ec2-user) is created. Use the "Key Pairs" menu option to create the associated public/private key for this account.



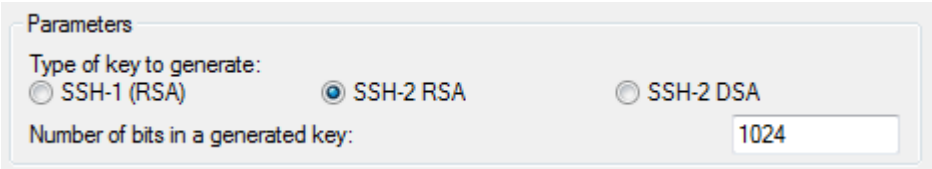

## 6.1 CONVERT THE PRIVATE KEY FOR USE WITH PUTTY

For users utilizing putty secure shell (SSH) client, the "PEM" file generated from AWS is not compatible with the putty application. In order to login, you must use the "puttygen" program to convert the private key.

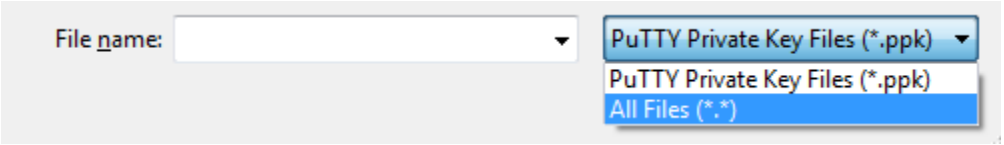
**NOTE:** The following instruction are taken from:

<< https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html?console_help=true>>

1. Start PuTTYgen (for example, from the **Start** menu, click **All Programs > PuTTY > PuTTYgen**).
2. Under **Type of key to generate**, select **SSH-2 RSA**.



3. Click **Load**. By default, PuTTYgen displays only files with the extension .ppk. To locate your .pem file, select the option to display files of all types.




4. Select your .pem file for the key pair that you specified when you launch your instance, and then click **Open**. Click **OK** to dismiss the confirmation dialog box.
5. Click **Save private key** to save the key in the format that PuTTY can use. PuTTYgen displays a warning about saving the key without a passphrase. Click **Yes**.

**SPECIAL NOTE:** A passphrase on a private key is an extra layer of protection, so even if your private key is discovered, it can't be used without the passphrase. The downside to using a passphrase is that it makes automation harder because human intervention is needed to log on to an instance, or copy files to an instance.Specify the same name for the key that you used for the key pair (for example, my-key-pair). PuTTY automatically adds the .ppk file extension.

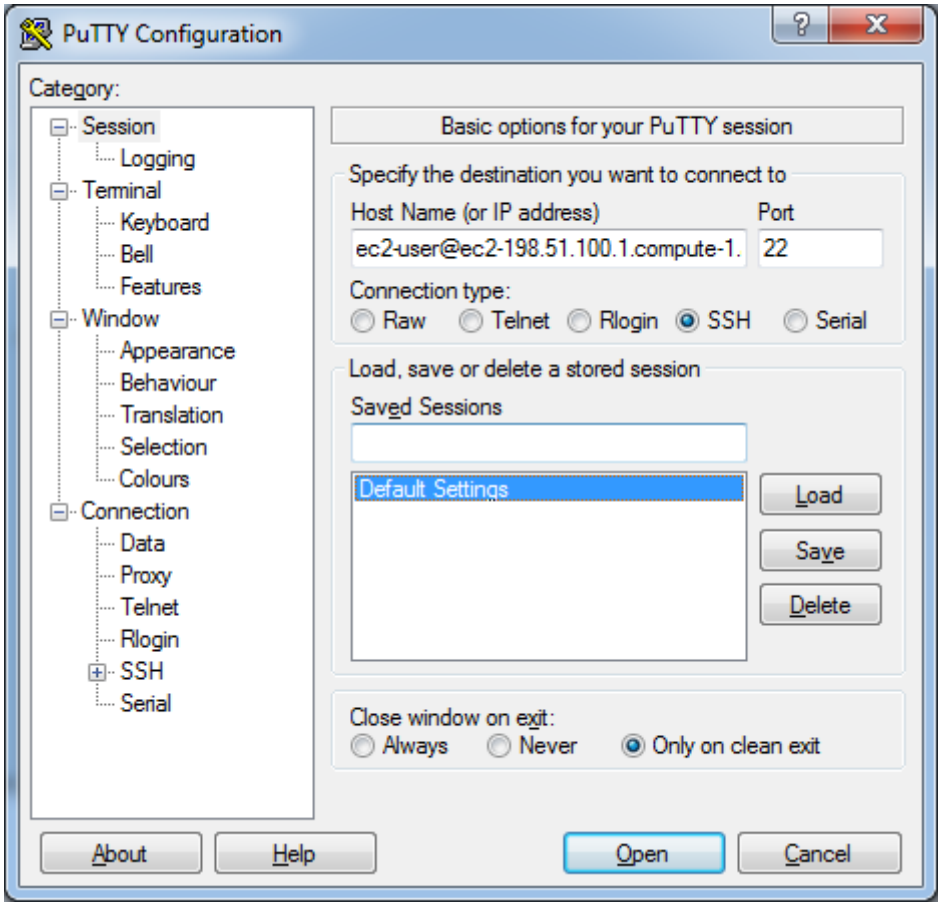
5. Your private key is now in the correct format for use with PuTTY. You can now connect to your instance using PuTTY's SSH client.

## 6.2 Starting a PuTTY Session

Use the following procedure to connect to your Linux instance using PuTTY. You'll need the `.ppk` file that you created for your private key.

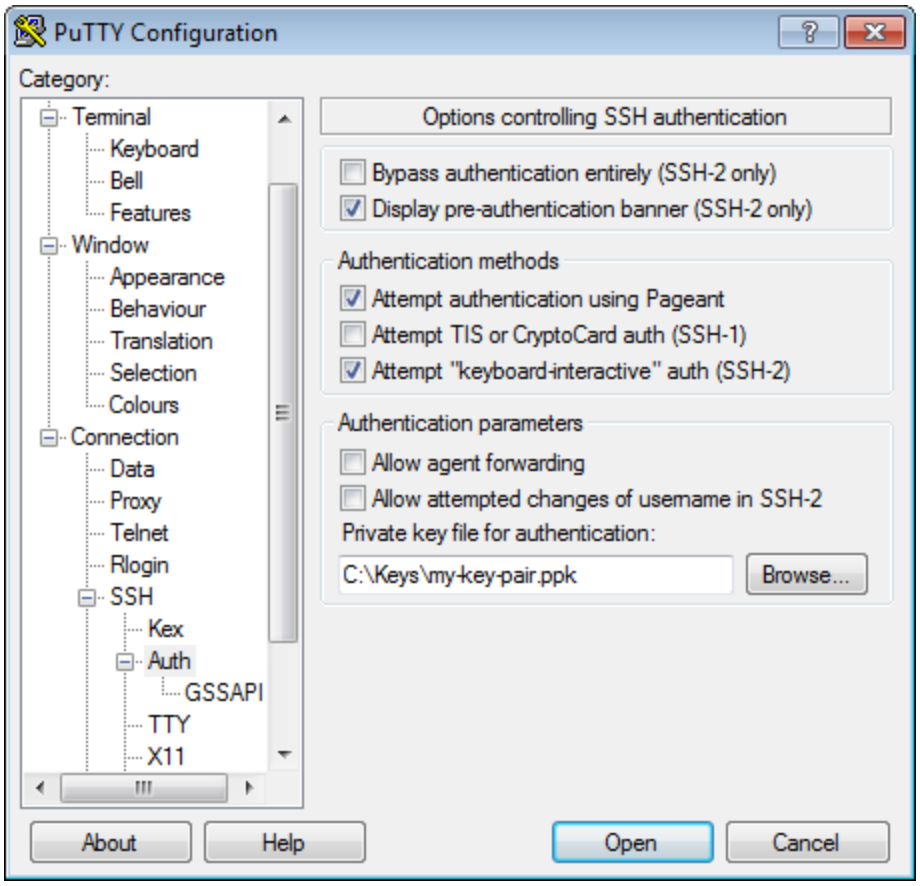
**To start a PuTTY session**

1. Start PuTTY (from the **Start** menu, click **All Programs > PuTTY > PuTTY**).
2. In the Category pane, select **Session** and complete the following fields:
   - In the **Host Name** box, enter *`user_name`*@*`public_dns_name`*. Be sure to specify the appropriate user name for your AMI. For example:
   - For an Amazon Linux AMI, the user name is `ec2-user`.
   - Under **Connection type**, select **SSH**.
   - Ensure that **Port** is 22.




3. In the **Category** pane, expand **Connection**, expand **SSH**, and then select **Auth**. Complete the following:

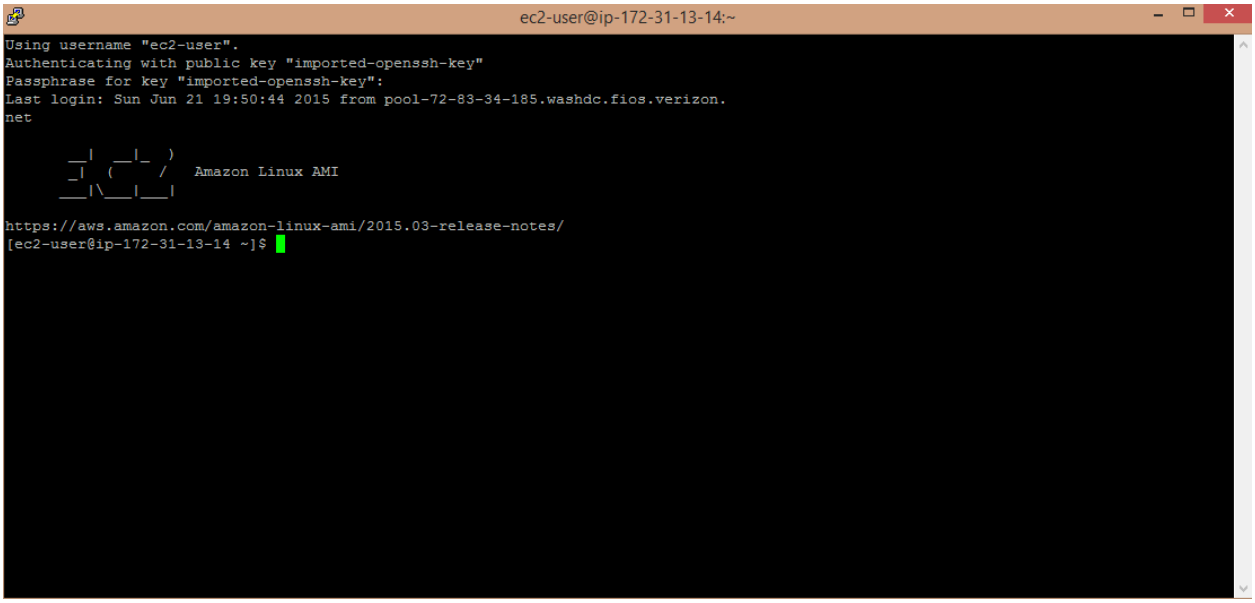
    - Click **Browse**.
    - Select the `.ppk` file that you generated for your key pair, and then click **Open**.
    - (Optional) If you plan to start this session again later, you can save the session information for future use. Select **Session** in the **Category** tree, enter a name for the session in **Saved Sessions**, and then click **Save**.
    - Click **Open** to start the PuTTY session.




4. If this is the first time you have connected to this instance, PuTTY displays a security alert dialog box that asks whether you trust the host you are connecting to.

5. Click **Yes**. A window opens and you are connected to your instance.

**NOTE:** If you specified a passphrase when you converted your private key to PuTTY's format, you must provide that passphrase when you log in to the instance.


ec2-user@ip-172-31-13-14:~
Using username "ec2-user".
Authenticating with public key "imported-openssh-key"
Passphrase for key "imported-openssh-key":
Last login: Sun Jun 21 19:50:44 2015 from pool-72-83-34-185.washdc.fios.verizon.
net
__|  __|_  )
_|  (     /   Amazon Linux AMI
___|___|___|
https://aws.amazon.com/amazon-linux-ami/2015.03-release-notes/
[ec2-user@ip-172-31-13-14 ~]$

**This Page intentionally Left Blank**

# 7.0 SERVER / APPLICATION CONFIGURATION

1. Update all software pakcages. Run the following command:

```
sudo yum update -y
```

2. Install Additional Packages. Run the following commands:

```
sudo yum install git -y
sudo yum install docker -y
```

## 7.1 CONFIGURE DOCKER

Run the following command to start the docker service

```
$sudo service docker start
```

Add the login user to the docker group to avoid having to use sudo for all docker commands.

```
$sudo usermod –a –G docker ec2-user
```

## 7.2 CONFIGURE PROTOTYPE

Create a certificate repository for the prototype server

```
$sudo mdir /var/certs
```

Place the following files into the /var/certs directory

| File | Description |
|---|---|
| server.crt | Digitially Signed Public SSL/TLS Key<br>Example:<br>-----BEGIN CERTIFICATE-----<br>MIIFfDCCBGSgAwIBAgIRANfJJe8ZwOReYSC…<br>-----END CERTIFICATE----- |
| server.key | Private Key SSL/TLS Key<br>Example:<br>----BEGIN PRIVATE KEY-----<br>MIIEwAIBADANBgkqhkiG9w0BAQEFAASCBKmYF4syN5nqL03lDVf++…<br>-----END PRIVATE KEY----- |
| server-ca.crt | Certificate Signing Chain<br>Example (one certificate per row):<br>MIIGCDCCA/…<br>MIIFdDCCBFygAwIBAgIQJ2buVutJ846r13Ci/…<br>MIIENjCCAx6gAwIBAgIBATANBgkqhkiG9w0… |
| api.key | OpenFDA API Key<br>Example (just the key on a single row):<br>Zh38TJcHtzt4gW4EfdY7dh… |

## 7.3 RUN DOCKER CONTAINERS

1. Use the following commands to start the Web and NodeJS Containers for supporting the OpenFDA Prototype from Intellect Solutions.

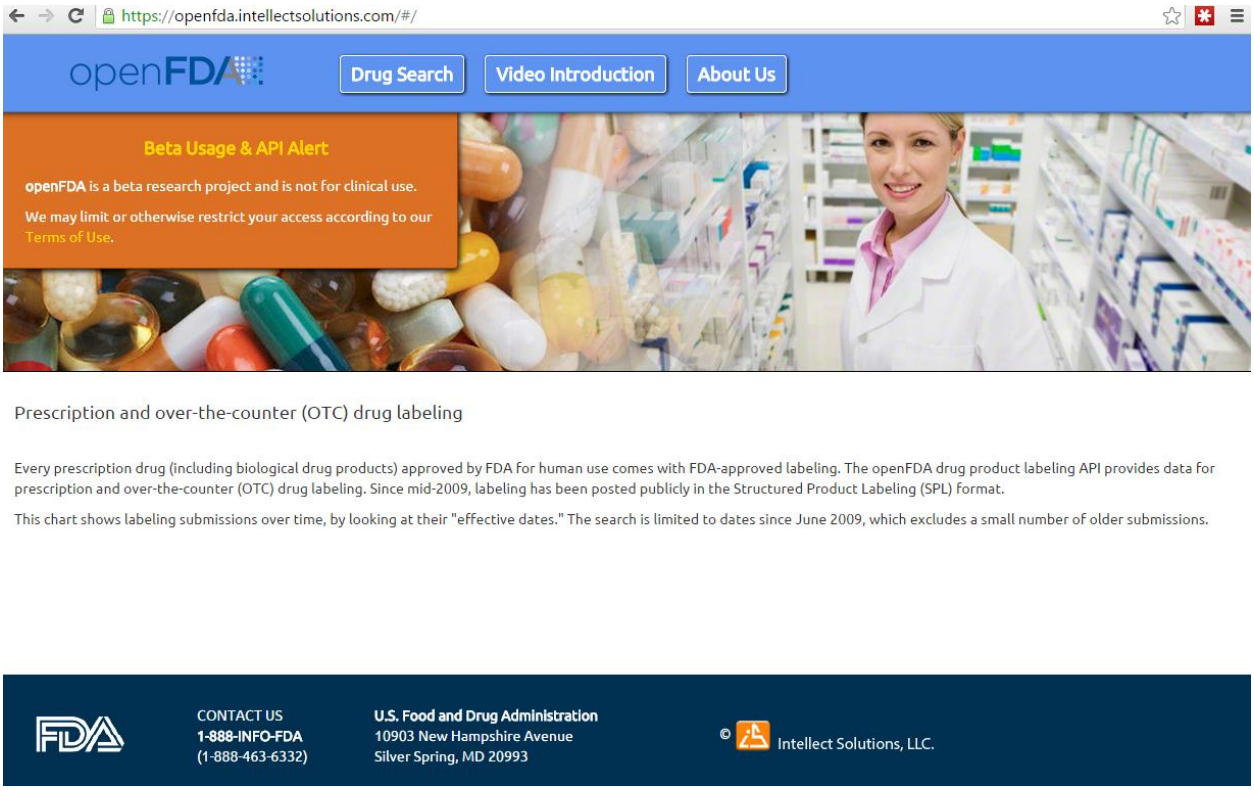
```
$docker run -p 8000:8000 -v /var/certs:/var/certs -d \
intellectsolutionsllc/openfda_nodejs

$docker run -p 80:80 -p 443:443 -v /var/certs:/var/certs -d \
intellectsolutionsllc/openfda_httpd
```

**Note:** The Docker Run command will automatically download the latest image from the public repository if it is not already on the system and will check for newer versions on each restart.

2. Use a browser to confirm the web application is deployed and validate against Interface Design Document (IDD)

```
https://openfda.intellectsolutions.com
```




3. Use a browser (recommend Chrome to see the resulting JSON response) to confirm the NodeJS application is deployed and validate against Interface Design Document (IDD)

```
https://openfda.intellectsolutions.com:8000/ openfda/listBrandNamePresDrugs
```


## 7.4 REBUILD DOCKER CONTAINERS FROM GITHUB REPOSITORY

The following section describes the procedures to buld the docker containers from the latest version of the GitHub Project Repository

### 7.4.1 STEP 1 – CREATE A LOCAL CLONE OF THE GITHUB REPOSITORY

Login to the server through SSH and execute the following commands:

```
git clone --depth 1 \
git://github.com/IntellectSolutions-GSA-Prototype/OpenFDA_Prototype.git
~/gitRepository/ -b 1.0
cd ~/gitRepository
git fetch && git reset --hard origin/1.0
```

### 7.4.2 STEP 2 – BUILDE THE DOCKER CONTAINER

Execute the following commands to build the NodeJS Container

```
cd ~/gitRepository/Back\ End/NodeJS\ Files
docker build --tag=openfda_nodejs .

cd ~/gitRepository/Front\ End/NodeJS\ Files
docker build --tag=openfda_httd .
```

**Note:** Authorized contributors should tag the docker image as "intellectsolutionsllc/openfda_nodejs" and "intellectsolutionsllc/openfda_httpd" respectively

### 7.4.3 STEP 3 – UPDATE THE DOCKER REPSOITORY (AUTHORIZED CONTRIBUTORS ONLY)

Authorized contributors can upload the resulting updated docker image using the following commands

```
docker login
docker push intellectsolutionsllc/openfda_nodejs
docker push intellectsolutionsllc/openfda_httpd
```

**Note:** Authorized contributors must provide a valid login credention (username, password, and email) to upload the containers to the Docker repository.

## 7.5 SETUP / CONFIGURE MONITORING

The following instructions and configuration changes were based on the documentation provided by Amazon at https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/mon-scripts.html

### 7.5.1 SETUP AWSLOGGING

```
sudo yum install awslogs
sudo vi /etc/awslogs/awscli.conf

[default]
region = <us-east-1, us-west-1, us-west-2, eu-west-1, eu-central-1, ap-southeast-1,
ap-southeast-2, or ap-northeast-1>
aws_access_key_id = <YOUR ACCESS KEY>
aws_secret_access_key = <YOUR SECRET KEY>
```

Note: Region must match the region the instance was created.
Update /etc/awslogs/awslogs.conf

```
[/var/log/messages]
file = /var/log/messages
buffer_duration = 5000
initial_position = start_of_file
datetime_format = %b %d %H:%M:%S
log_stream_name = {instance_id}
log_group_name = /var/log/messages

[/var/log/secure]
datetime_format = %b %d %H:%M:%S
file = /var/log/secure
buffer_duration = 5000
log_stream_name = {instance_id}
initial_position = start_of_file
log_group_name = /var/log/secure
encoding = utf-8

[/var/log/httpd/ssl_request_log]
datetime_format=%d/%b/%Y:%H:%M:%S
file = /var/log/httpd/ssl_request_log
buffer_duration = 5000
log_stream_name = {instance_id}
initial_position = start_of_file
log_group_name = /var/log/httpd/ssl_request_log
encoding = utf-8
```

Restart Logging

```
sudo service awslog restart
```

### 7.5.2 SETUP SYSTEM MONITORING

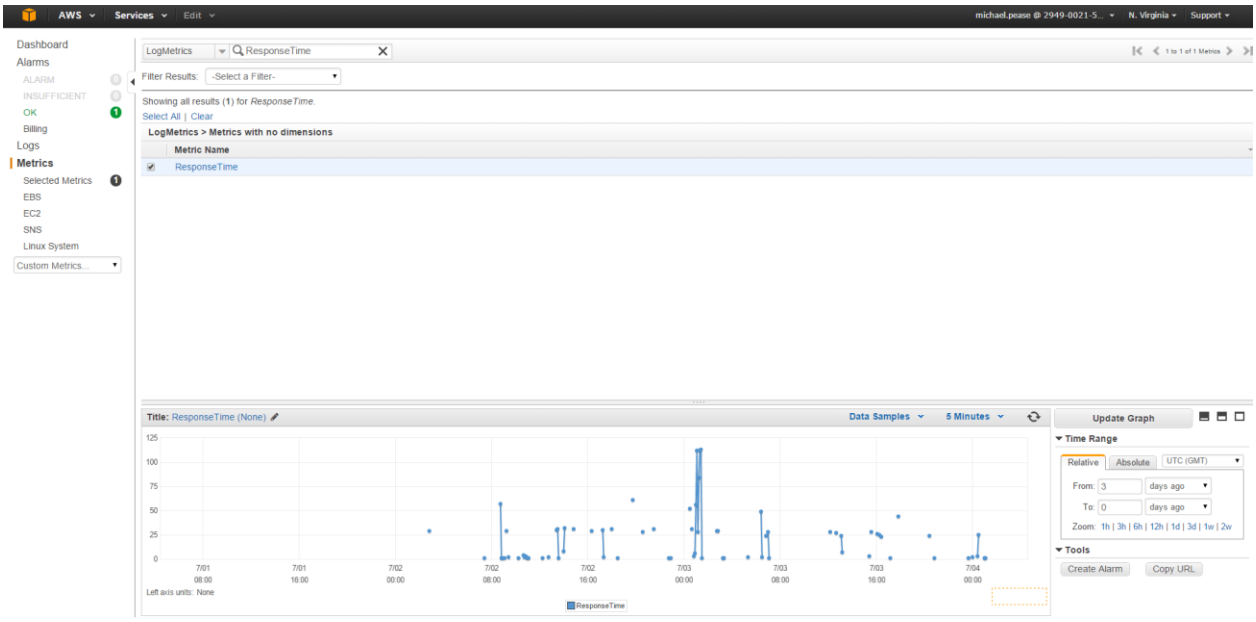
```
sudo yum install perl-DateTime perl-Sys-Syslog perl-LWP-Protocol-https
wget http://aws-cloudwatch.s3.amazonaws.com/downloads/CloudWatchMonitoringScripts-1.2.1.zip
unzip CloudWatchMonitoringScripts-1.2.1.zip
rm CloudWatchMonitoringScripts-1.2.1.zip

crontab -e
# To Run Every 5 Minutes
*/5 * * * * ~/aws-scripts-mon/mon-put-instance-data.pl --mem-util --disk-space-util --disk-path=/ --from-cron
```

### 7.5.3 AWS MONITORING

Once the procedures outlined in this section are completed, the logs and statistics will become available thorugh the AWS CloudWatch dashboard and allow ou to monitor over time, key performance metrics associated with the server and application performance.

The image below is the web server response times in microseconds.



# APPENDIX A ACRONYMS AND ABBREVIATIONS

Table A- 1 lists the Acronyms used in this document.

**Table A- 1: Acronyms and Abbreviations List**

| Acronym | Definition |
|---|---|
| AS | Application Server |
| CI | Configuration Item |
| DISA | Defense Information Systems Agency |
| EA | Enterprise Architect; Enterprise Architecture |
| EF | Expeditionary Framework |
| EM | Enterprise Manager |
| EMSS | Enterprise Master Security Server |
| FEP | Front End Processor |
| ID | Identifier |
| IDD | Interface Design Document |
| IE | Interface Engine |
| LCS | Local Cache Server |
| OS | Operating System |
| RTM | Requirements Traceability Matrix |
| S&I | Summary and Impact |
| SDD | Software Design Document |
| SIG | System Installation Guide |
| SP | Service Pack |
| SRS | System Requirements Specification |
| SSDD | System Subsystem Design Document |
| SSH | Secure Shell |
| STP | Software Test Plan |
| STR | Software Test Results |
| TCP/IP | Transmission Control Protocol/Internet Protocol; Transport Control Protocol/Internet Protocol |
| VDD | Version Description Document |
| WSE | Web Service Enhancements |

**This Page intentionally Left Blank**