Project RF Installation

The frame work I created works with LAMP (Linux, Apache, MySQL, PHP). The easiest way to get up and running is to download and install VirtualBox (http:// http://www.virtualbox.org/wiki/Downloads) or VMWare Player (http:// www.vmware.com/go/downloadplayer) and obtain a virtual appliance. The appliance that I have used is the LAMP Stack Appliance from TurnKey (http://www.turnkeylinux.org/lamp).

When you first boot the appliance you will be asked to set the system root and MySQL root passwords. You can access webmin and phpadmin by connecting to the server via a browser. You will need to log into MySQL and create a database. I called mine projectRF.

```
#mysql –u root –p
Enter Password:
mysql>create database projectRF;
mysql>grant usage on *.* to root@localhost identified by
'<your_password>';
mysql>quit
```

Download the framework and use the provided SQL file to load the database with the proper tables.

```
#wget http://www.jedge.com/docs/projectRF.zip
#unzip projectRF.zip
#cd projectRF
#mysql -u root -p projectRF < projectRF.sql
Enter Password:
```

After you enter the password the database will be setup. Next we will copy the PHP files to the Apache webroot.

```
#rm projectRF.sql
#cd ..
#mv projectRF /var/www
#chmod 777 /var/www/projectRF/nessus/csvfiles
#chmod 777 /var/www/projectRF/nmap/csvfiles
#chmod 777 /var/www/projectRF/kismet/kmlfiles
```

You will then have to edit the PHP configuration file and tell it how to connect to the new database.

```
#vim /var/www/projectRF/main/config.php
```

Instructions are provided in the config.php. They are pretty self-explanatory. All you need to change in the file is the password you set for the MySQL database. Next you will have to install DB.php as this is used by the framework to interact with the database.

```
#apt-get update
#apt-get autoremove
#apt-get install php-db php5-gd
```

Accept all defaults.

Lastly you may have to modify the php.ini file to accept larger upload file sizes and increase the maximum execution time. The default configuration allows for a total POST size of 16M, a max file upload size of 8M, and a maximum execution size of only 30 seconds. This is insufficient for large scans that require a lot of parsing and report generation time.

The variables post_max_size (line 728), upload_max_filesize (line 879), max_execution_time (line 440), and max_input_time (line 450) should be modified to allow for larger XML files. By default the VIM install on the appliance does not show line numbers. When you first enter VIM you should enter a colon (:) and hit enter. Then type the command `set number` and hit enter. You will now have line numbers. Google is your friend for finding out how to use VIM.

```
#vim /etc/php5/apache2/php.ini
```

I set each file size value to 1G and the execution time to 1800 (30 minutes). I've parsed 1G WebInspect and AppDetective files that large without issue.

Restart the Apache webserver to load the modified php.ini file
```
#/etc/init.d/apache2 restart
```

You can now access the framework from via the browser. You can now upload your Nessus XML files and start generating reports!

Frequent Error Messages

E:

PHP Fatal error: Maximum execution time of 30 seconds exceeded in /var/www/projectRF/nessus/executiveReport.php on line 481

A:

The maximum execution time variable (max_execution_time) in the php.ini file needs to be increased.