This document describes how you can set up an Nginx Load Balancer for FirstClass Web Services. Nginx will be running on CentOS, but the instances of fcws can either be running on the same machine, or on another Mac OR Linux box.

For the purposes of this document, I will demonstrate both Nginx and 3 instances of FCWS running under CherryPy all on the same box.

Why Load Balance?

Load balancing is especially important for networks where it's difficult to predict the number of requests that will be issued to a server. Such is the case with FirstClass Web Services. If you find that responses are slow, you may find that you have reached a threshold on the CherryPy instance that is running FirstClass Web Services. Each user uses at least 2 threads so if you have a higher concurrency, you should consider installing a load balancer in front of multiple instances of FCWS, each set to provide 200 threads. This will mean that the load will be distributed across each of the instances and each instance can handle 100 users.

There are many different types of load balancers, including Apache and F5, but Nginx is the one that we are going to install.

The free version of Nginx that we will install and compile has the advantage of supporting Web Sockets, whereas Apache does not.

Multiple Instances of FCWS

You can have multiple instances of fcws running under CherryPy on the same Mac or Linux box. In this section we will learn how to modify the startup script to initialize three instances running on ports 8000, 8001 and 8002.

The startfcws.command or startfcws.sh startup script

The initial startup script for fcws is located in the FirstClass Web Services folder and has a default that looks something like this:

Macintosh

```
sudo /System/Library/Frameworks/Python.framework/Versions/2.7/Resources/Python.app/
Contents/MacOS/Python "/Library/FirstClass Web Services/fcws-12.1.0.025N-osx/fcws/cherry.
pyc" -H 0.0.0.0 -p 80 -e True
```

Linux

```
sudo /usr/bin/python2.6 "/opt/FirstClass Web Services/fcws-12.1.0.025N-linux/fcws/cherry.
pyc" -H 0.0.0.0 -p 80 -e True
```

The fcws version number will change with each upgrade. The above example is showing fcws 12.1 build 25.

Modify the startsfcws script to start 3 different instances

We are going to modify the start script to listen on 3 different **http** ports. We will **not** be listening on https as we will let Nginx do that. We can place our fcws servers behind a firewall and not allow access to the http ports except from the nginx box itself.

To create 3 instances of fcws to listen for http only on 3 different ports, we can modify the startup script as follows:

Macintosh

```
sudo /System/Library/Frameworks/Python.framework/Versions/2.7/Resources/Python.app/
Contents/MacOS/Python "/Library/FirstClass Web Services/fcws-12.1.0.025N-osx/fcws/cherry.
pyc" -H 0.0.0.0 -p 8001 --httpthreads=200 &
/System/Library/Frameworks/Python.framework/Versions/2.7/Resources/Python.app/Contents/
MacOS/Python "/Library/FirstClass Web Services/fcws-12.1.0.025N-osx/fcws/cherry.pyc" -H 0.
0.0.0 -p 8002 --httpthreads=200 &
/System/Library/Frameworks/Python.framework/Versions/2.7/Resources/Python.app/Contents/
MacOS/Python "/Library/FirstClass Web Services/fcws-12.1.0.025N-osx/fcws/cherry.pyc" -H 0.
0.0.0 -p 8003 --httpthreads=200
```

Linux

```
sudo /usr/bin/python2.6 "/opt/FirstClass Web Services/fcws-12.1.0.025N-linux/fcws/cherry.
pyc" -H 0.0.0.0 -p 8001 --httpthreads=200 &
/usr/bin/python2.6 "/opt/FirstClass Web Services/fcws-12.1.0.025N-linux/fcws/cherry.pyc" -
```

```
H 0.0.0.0 -p 8002 --httpthreads=200 &
/usr/bin/python2.6 "/opt/FirstClass Web Services/fcws-12.1.0.025N-linux/fcws/cherry.pyc" -
H 0.0.0.0 -p 8003 --httpthreads=200
```

Notice that in both cases, i have done the following:

- removed the -e True switch as we will not be enabling https here
- set the ports to 8001, 8002 and 8003 and
- set each instance to have 200 http threads.

After starting fcws, the console will display something that looks like this, indicating that all three instances are running.

```
[fcadmin@localhost ~]$ fcwsctl start
[sudo] password for fcadmin:
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]WARNING: HTML sanitizer disabled. Python LXML-CLEANER package not found.
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]WARNING: HTML sanitizer disabled. Python LXML-CLEANER package not found.
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]Running Python 2.6.6 (r266:84292, Jan 22 2014, 09:42:36)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-4)]
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]Configuring CherryPy Server for hosting OpenText FC WebServer 12.1.0.026 (FC WebAPI 1) (Linux).
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]WARNING: Installed version of Python does not support TlSv1.2 protocol. Switched to TLSv1
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]ERROR: CherryPy WSGI Server - SSL Key file (key.pem) could not be found. SSL server will not be configured.
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]CherryPy WSGI Server - Configuring 0.0.0.0 on HTTP port 8003, running 200 threads.
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]Running Python 2.6.6 (r266:84292, Jan 22 2014, 09:42:36)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-4)]
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]Configuring CherryPy Server for hosting OpenText FC WebServer 12.1.0.026 (FC WebAPI 1) (Linux).
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]WARNING: Installed version of Python does not support TlSv1.2 protocol. Switched to TLSv1
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]ERROR: CherryPy WSGI Server - SSL Key file (key.pem) could not be found. SSL server will not be configured.
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]CherryPy WSGI Server - Configuring 0.0.0.0 on HTTP port 8002, running 200 threads.
[18/Jul/2015:07:24:32] ENGINE Bus STARTING
[18/Jul/2015:07:24:32] ENGINE Bus STARTING
[18/Jul/2015:07:24:32] ENGINE Started monitor thread '_TimeoutMonitor'.
[18/Jul/2015:07:24:32] ENGINE Started monitor thread '_TimeoutMonitor'.
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]WARNING: HTML sanitizer disabled. Python LXML-CLEANER package not found.
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]Running Python 2.6.6 (r266:84292, Jan 22 2014, 09:42:36)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-4)]
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]Configuring CherryPy Server for hosting OpenText FC WebServer 12.1.0.026 (FC WebAPI 1) (Linux).
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]WARNING: Installed version of Python does not support TlSv1.2 protocol. Switched to TLSv1
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]ERROR: CherryPy WSGI Server - SSL Key file (key.pem) could not be found. SSL server will not be configured.
Sat Jul 18 07:24:32 2015 127.0.0.1 [system]CherryPy WSGI Server - Configuring 0.0.0.0 on HTTP port 8001, running 200 threads.
[18/Jul/2015:07:24:32] ENGINE Bus STARTING
[18/Jul/2015:07:24:32] ENGINE Started monitor thread '_TimeoutMonitor'.
[18/Jul/2015:07:24:32] ENGINE Serving on 0.0.0.0:8002
[18/Jul/2015:07:24:32] ENGINE Bus STARTED
[18/Jul/2015:07:24:32] ENGINE Serving on 0.0.0.0:8003
[18/Jul/2015:07:24:32] ENGINE Bus STARTED
[18/Jul/2015:07:24:32] ENGINE Serving on 0.0.0.0:8001
[18/Jul/2015:07:24:32] ENGINE Bus STARTED
/usr/sbin/fcwsctl start: FirstClass Web Services (cherry.py) started ( 2651 2652 2653 2654 ).
[fcadmin@localhost ~]$
```

Nginx for CentOS

Nginx is a free, open-source, high-performance HTTP server and reverse proxy, as well as an IMAP/POP3 proxy server. Nginx is known for its high performance, stability, rich feature set, simple configuration, and low resource consumption.

Nginx powers several high-visibility sites, such as Netflix, Hulu, Pinterest, CloudFlare, Airbnb, WordPress.com, GitHub, SoundCloud, Zynga, Eventbrite, Zappos, Media Temple, Heroku, RightScale, Engine Yard and MaxCDN

In this document, I will be installing Nginx for **CentOS** using a modified script provided by Kevin Worthington http://kevinworthington.com/nginx-for-mac-os-x-mavericks-in-2-minutes/ The script compiles a basic version of Nginx 1.5.7.

The script accompanies this tutorial.

The script has been tested on **CentOS 6.6**.

Installation Prerequisites

Before you can use the script to install Nginx on a minimal install CentOS installation you must first ensure that the CentOS **Development tools** are installed along with **wget** and **openssl**. You can do this by logging in to your CentOS box as **root** and issuing the following commands in terminal.

Installing CentOS Development Tools.

First clear the yum cache

```
[root@localhost ~]# yum clean all
```

now install the developer tools. This process could take several minutes.

```
[root@localhost ~]# yum groupinstall "Development tools"
```

Installing wget

We will be retrieving some files using wget which is not normally installed by default on CentOS. We can install it though, by issuing the following command in terminal while logged in as root.

```
[root@localhost ~]# yum install wget
```

Installing OpenSSL

Finally, you need to install OpenSSL as this library will be used by Nginx to serve secure web pages. To install OpenSSL, issue the following command in terminal while logged in as root.

```
[root@localhost ~]# yum install openssl openssl-devel
```

Installing Nginx

Once you have the prerequisites installed, you can use the provided script to download and compile Nginx.

To utilize the script you must use terminal to navigate to the folder where you have downloaded the shell script named **build-nginx-CentOS.sh**

Before you can execute the script, you first have to change the permissions. While logged is as root with terminal open and in the same directory as the script, type in the following command.

```
[root@localhost ~]# chmod a+x build-nginx-CentOS.sh
```

Now you can execute the script using the command

```
[root@localhost ~]# ./build-nginx-CentOS.sh
```

This will download, compile and start **nginx** for your CentOS installation.

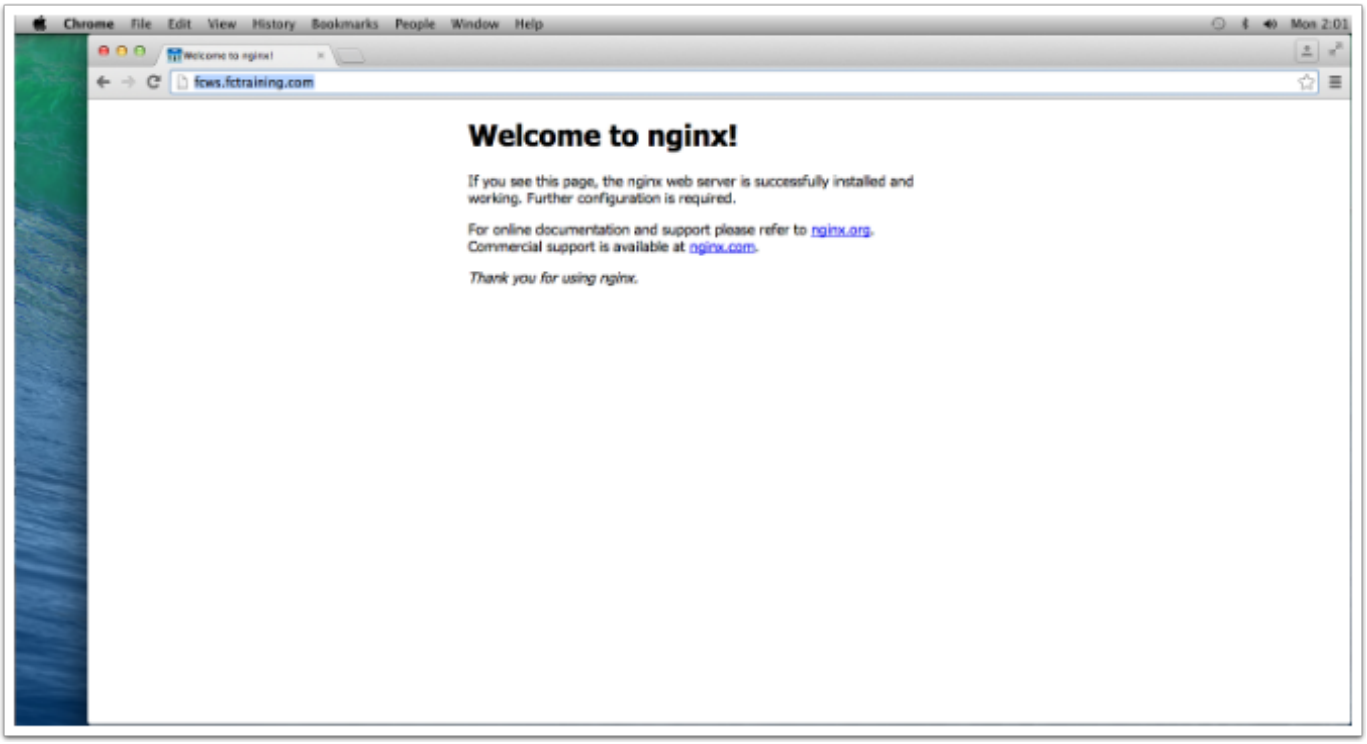
Testing the Install

If all goes well, Nginx will have been installed and started.

You can test your server by visiting your server by either using localhost on the server where you have installed Nginx or, use the domain name that you have established for the IP where you have installed Nginx.

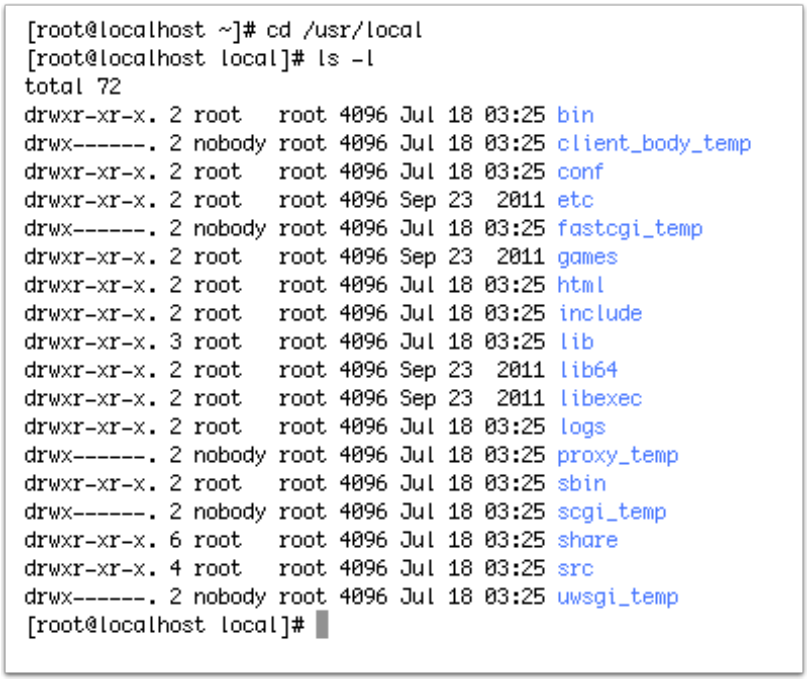
In my case, Nginx is installed at fcws.fctraining.com

If you visit your site and see the following page, you know you have a successful installation



Nginx Directories

Nginx has been installed in the **/usr/local directory.**



Starting and stopping Nginx

We will have to stop Nginx and change the configuration file to make it a load balancer. You can use Terminal to stop it but you will have to use the full path in your command. The nginx executable is stored in the /usr/local/sbin folder so to stop the currently running nginx, type in the following in terminal:

```
sudo /usr/local/sbin/nginx -s stop
```

Adding Nginx as a system service

In this section, we will create a script that will transform the Nginx daemon into an actual system service. This will result in mainly two outcomes—the daemon will be controllable using standard commands, and more importantly, it will automatically be launched on system startup and stopped on system shutdown

Installing the init script

The first thing we need to do is create a shell script for starting and stoping the Nginx daemon.

1. First, navigate to the **/etc/init.d** directory and create a file called **nginx.** You can use your editor of choice.
2. The script for the init script has been provided as part of the resources for this tutorial. Copy and paste in the text from the **NginxService.txt** script into your nginx file and save.
3. Make the script executable by changing the permissions. To to this, issue this command in terminal

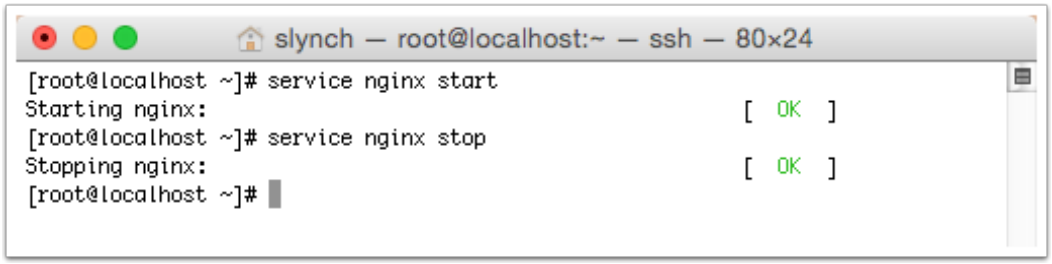
```
[root@localhost ~]# chmod +x /etc/init.d/nginx
```

At this point, you should already be able to start the service by typing `service nginx start` to start nginx and `service nginx stop` to stop it.

The last step here will be to make it so the script is automatically started at the proper runlevels. While logged in as root, open terminal and type:

```
[root@localhost ~]# chkconfig nginx on
```

Now when you boot your computer, nginx will automatically start.

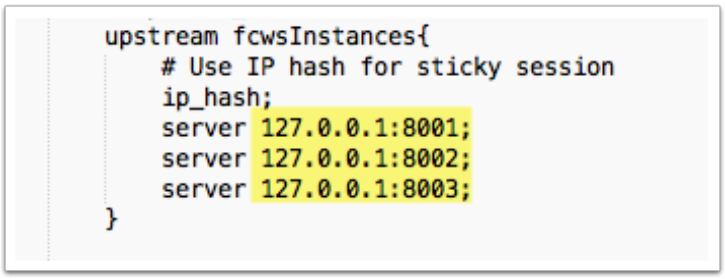


Updating the configuration file

I have provided you with a configuration file called **nginx.conf** that you can use to replace the one that is located in the **/usr/local/conf** folder. Rename the current file in that folder and copy the replacement one to that location.

Open the file in a text editor and confirm the following:

1. In the upstream fcwsInstances section, ensure that the server IP addresses and ports for your fcws instances are correct. In the case shown in the default, all three instances of fcws are running on the same box (127.0.0.1) but they could be running on a linux box somewhere else behind the firewall. The three ports are the ones that we configured (8001,8002 and 8003)



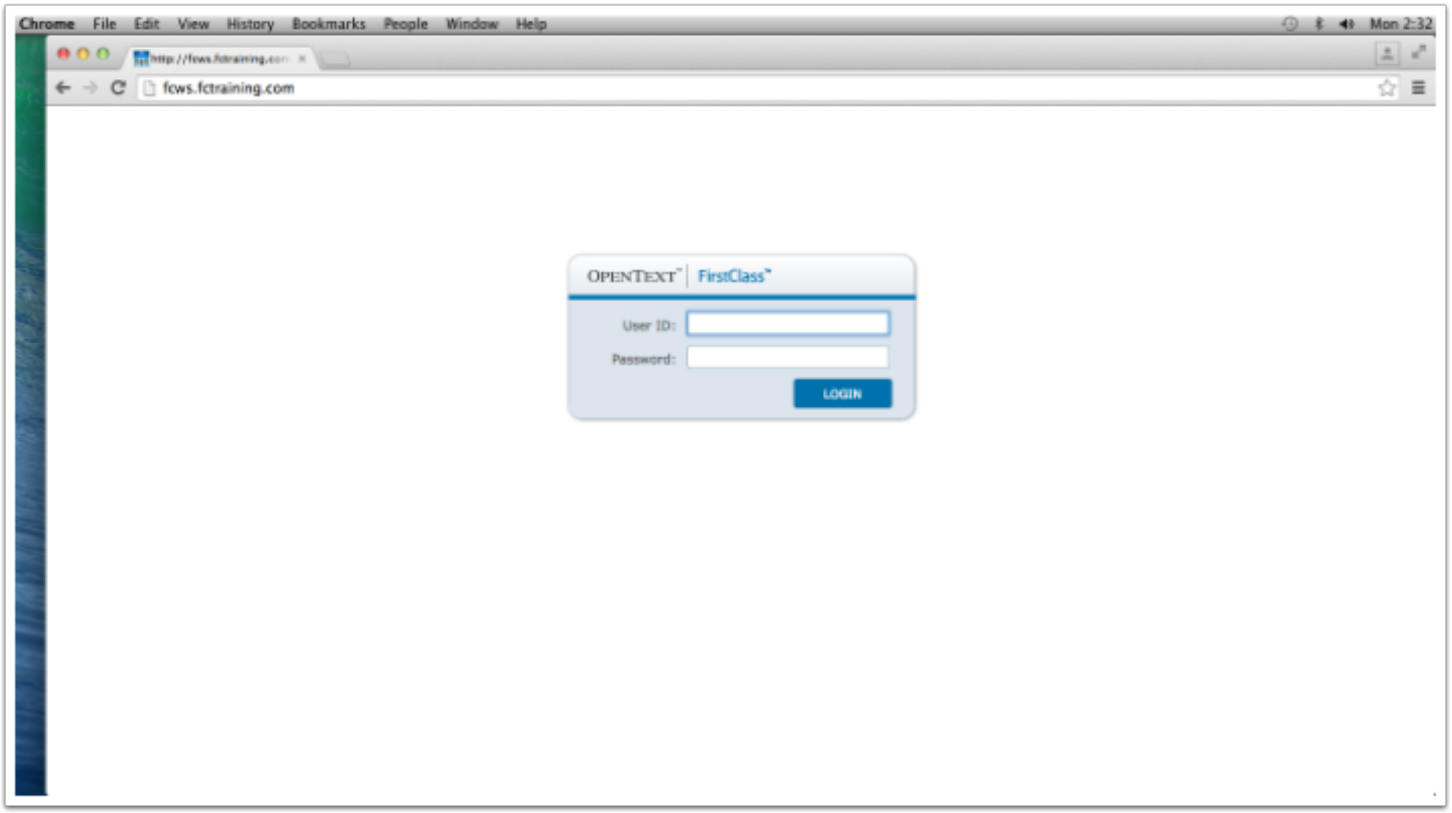
Start fcws and Nginx and test

If they are not already running, start your instances of fcws.

Start nginx using the command (assuming you have updated your environment path.

```
[root@localhost Installers]# service nginx start
```

If there are no errors, you should now be able to visit your nginx page and it will be directed to one of your fcws instances.



Installing SSL certificates

If you want to install ssl certificates, you can do this for nginx and modify the configuration file. But first, you will have to stop fcws and modify one of the parameters in the fcws.cfg file to disable the requirement for SSL certificates to be installed on fcws. The assumption here is that we will enable only

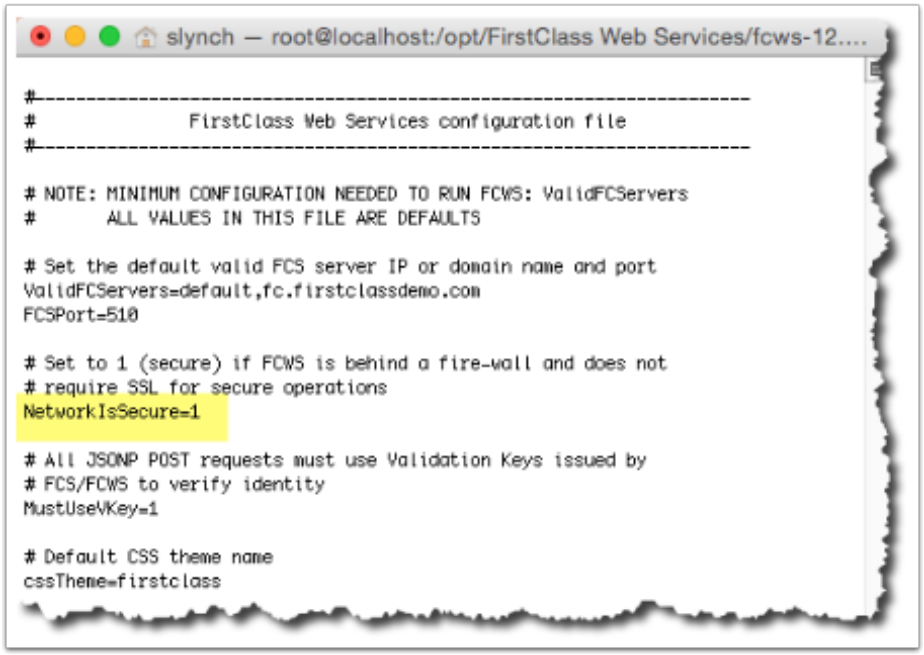
port 443 on the firewall to access the Nginx server and the three ports for fcws will only be accessible by the Nginx proxy behind the firewall and not accessible to the outside.

Modify fcws.cfg

Make sure you stop both fcws and Nginx and navigate to the fcws folder for your operating system.

Open **fcws.cfg** and modify the one line NetworkIsSecure=**0** to read NetworkIsSecure=**1**

Save and close the file and **restart fcws**.



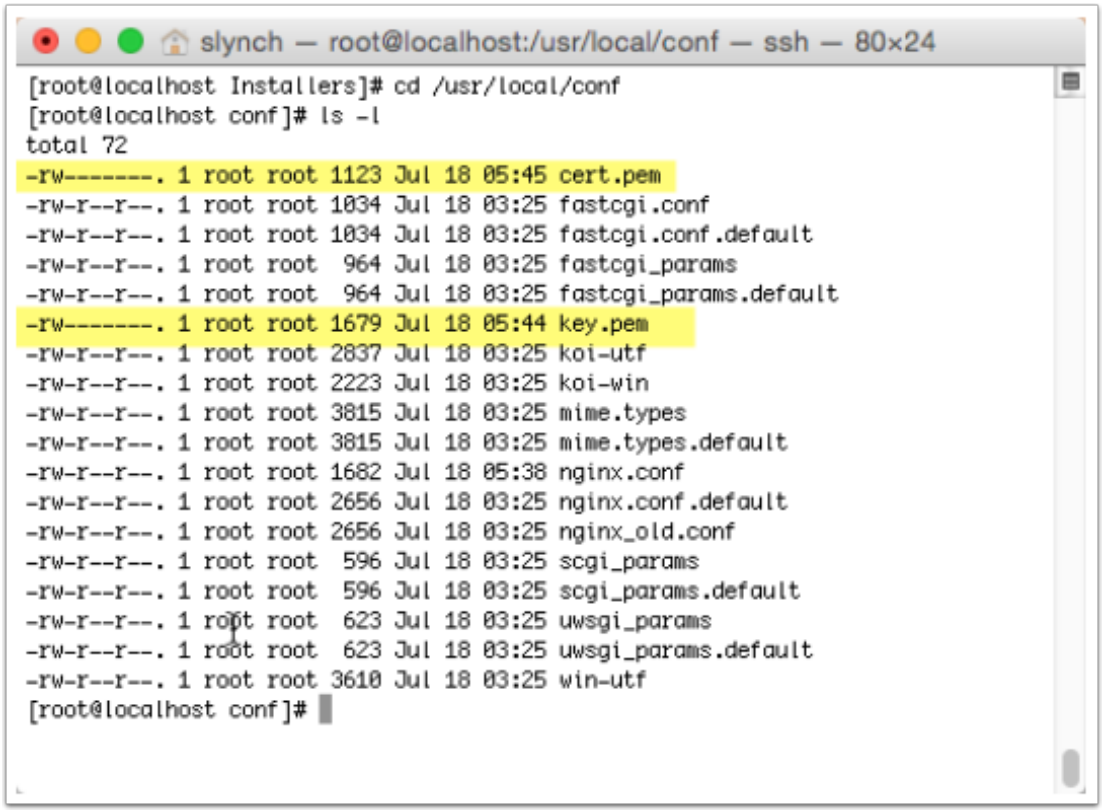
Adding Certificate Files to Nginx

Nginx requires both the private key and certificate files to be installed in the /usr/local/conf folder. These files can be any name, but the names will have to be referenced in the nginx.conf file in the next step. In keeping with the tradition set with a single fcws install, I have called my two files key.pem and cert.pem.

Copy both of these files in to the **/usr/local/conf** folder

NOTE: If your issuing agent supplies you with an intermediate certificate, this will have to be concatenated to the cert.pem file as there is no provision for referencing other certificates in the chain.

You can review this article from Comodo to see how to do this. https://support.comodo.com/index.php?/Default/Knowledgebase/Article/View/789/37/certificate-installation-nginx



Modify the ngnix.conf file

You can now modify the **nginx.conf** file that we have moved into the **/usr/local/conf** folder. Open it in a text editor and in the **server** section:

1. Uncomment lines 24 - 27 by removing the # at the beginning of the line. This will redirect all http traffic on port 80 to port 443, the secure port
2. Comment line 30 by adding a # in front of the line so that it does not listen on port 80 in this section
3. Remove the # from line 31 so that it now listens on port 443
4. Remove the # from lines 32 - 37 to enable the certificates
5. Ensure that your certificate and key file names match yours

Save the file

Test

Restart nginx using the following command in Terminal

```
[root@localhost conf]# service nginx start
```

Point your browser to the insecure port (in my case http://fcws.fctraining.com) and test that it redirects to **https** and recognizes the SSL certificate.