

Run a Bitnami Ubuntu 18.04 LTS HHVM stack on Windows 10 using Windows Subsystem for Linux

This tutorial explains how to set up and run a **Bitnami Ubuntu 18.04 Linux, Apache, mySQL, and HHVM (LAMH) stack** on a **Windows 10** machine by using the **Windows Subsystem for Linux (WSL)**. The use of **WSL** allows **Linux** to run alongside **Windows 10** without the need for a second device or virtual machine. Setting up the **LAMH** stack this way offers three significant benefits to the user.

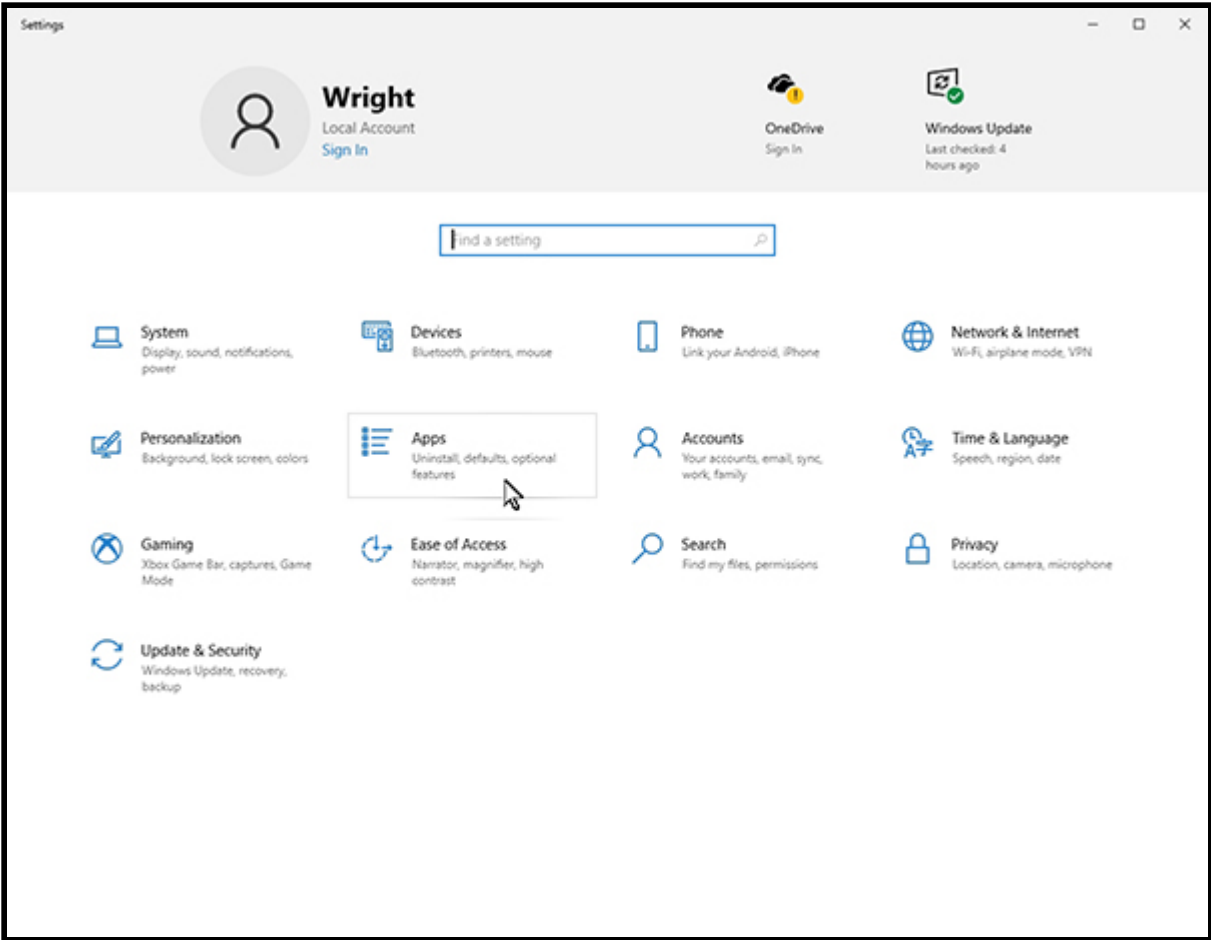
The first benefit is that the setup provides a nice development environment for a **LAMH** stack on a **Windows** machine. So for example, some change can be made to the stack or its contents and the result can be seen at the same time in a **Chrome** browser running on the **Windows** "machine". The user has access to the **Linux** command console within a window on the same screen.

The second benefit is that this approach creates an easy to reload or copy LAMH installation. By exporting a tarball archive backup of the setup it can easily be restored or reloaded to this baseline setup. Moreover, the tarball can be transported or copied to another **Windows 10** machine and made to run the same setup on the target computer. So for example all the students in a class room could easily have identically the same baseline setup. Here at **The Palace** the same setup is running on a workstation and on a laptop.

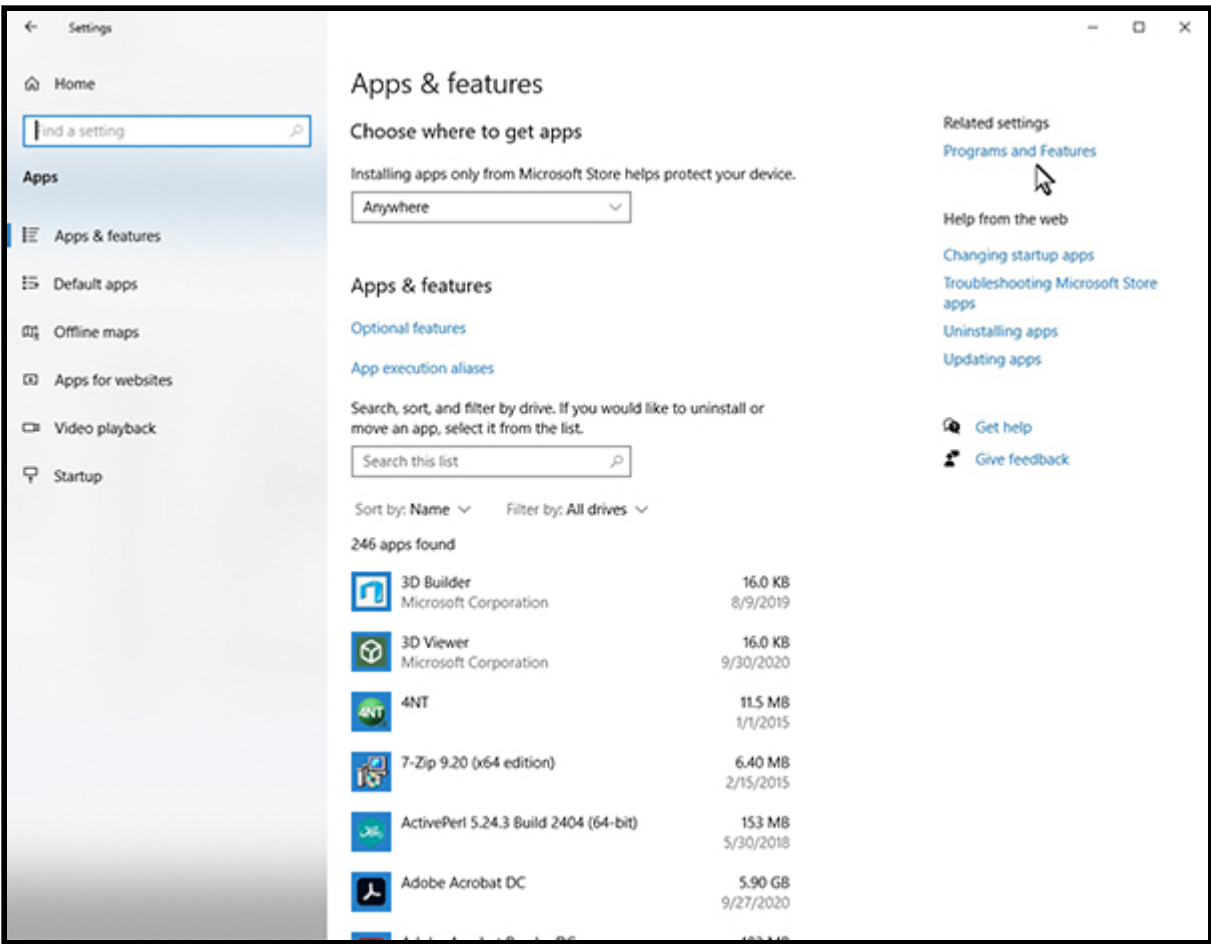
The third benefit is that using the **Bitnami Ubuntu HHVM stack** in this environment provides a close approximation to what is presented by the same stack installed on **Amazon Web Services (AWS)** or a true standalone server.

Before beginning the setup you need to make certain that your **Windows 10** version is up to date. If the version is current the rest of the setup should go without a hitch. It is also probably helpful to glance over the entire procedure as described here before beginning, and view the quick introductory video. All of the steps are relatively straightforward and none of this is rocket appliances.

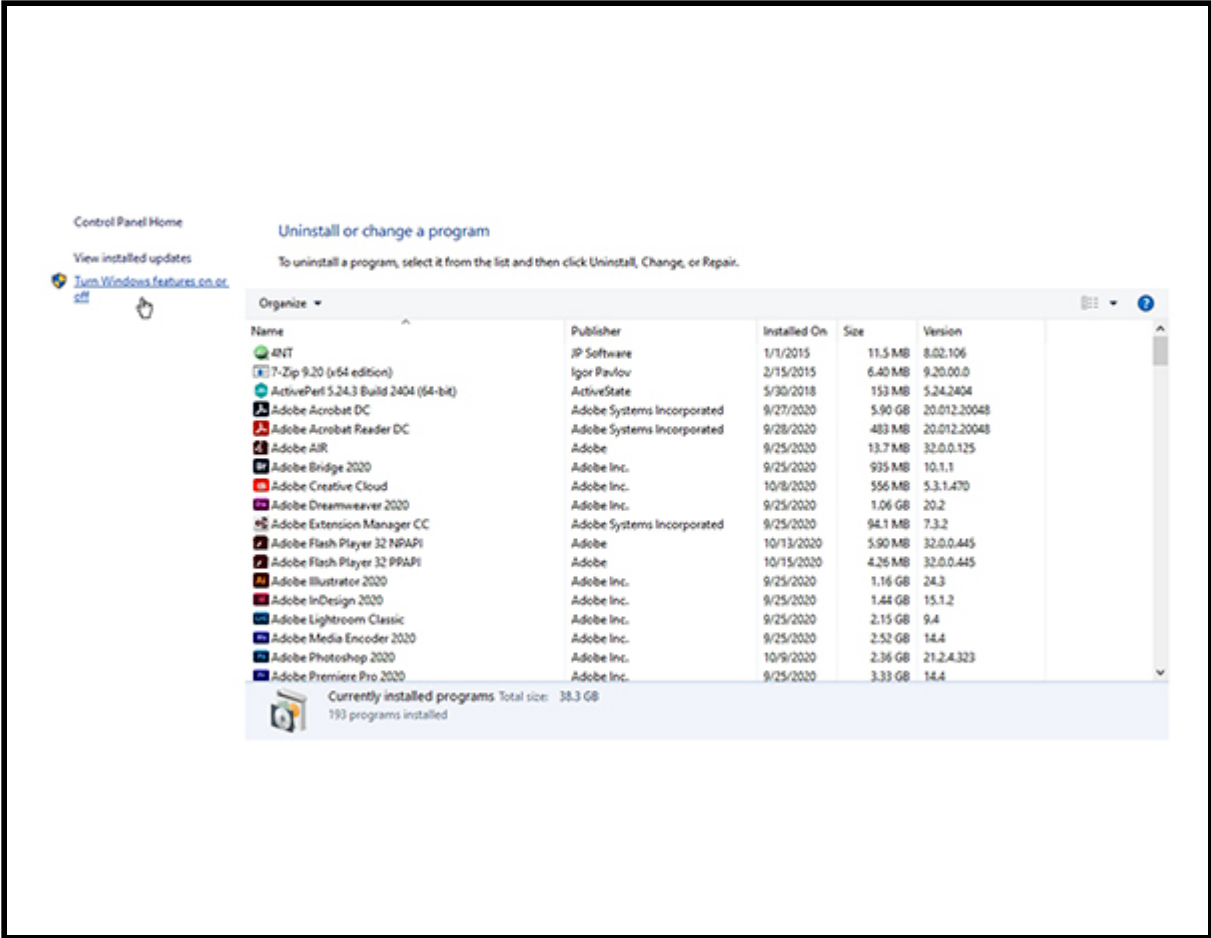
The first thing we need to do is install the **Windows Subsystem for Linux**, or **WSL**. **Microsoft** refers to the product we will use as the **WSL 2 lightweight utility virtual machine**. The method we use here is through the **Settings** dialog on the **Windows Start Menu**. Open the **Settings** main panel and click on **Apps**. From the **Apps & features** panel click **Programs and Features** on the right side. On the panel that opens click the **Turn Windows features on or off** option in the left pane. In the resulting dialog box locate the **Windows Subsystem for Linux** selection and select it and click **OK**. Finally click on the **Restart now** button. This completes the basic install of **WSL**. The following images show a visual recap of the process.



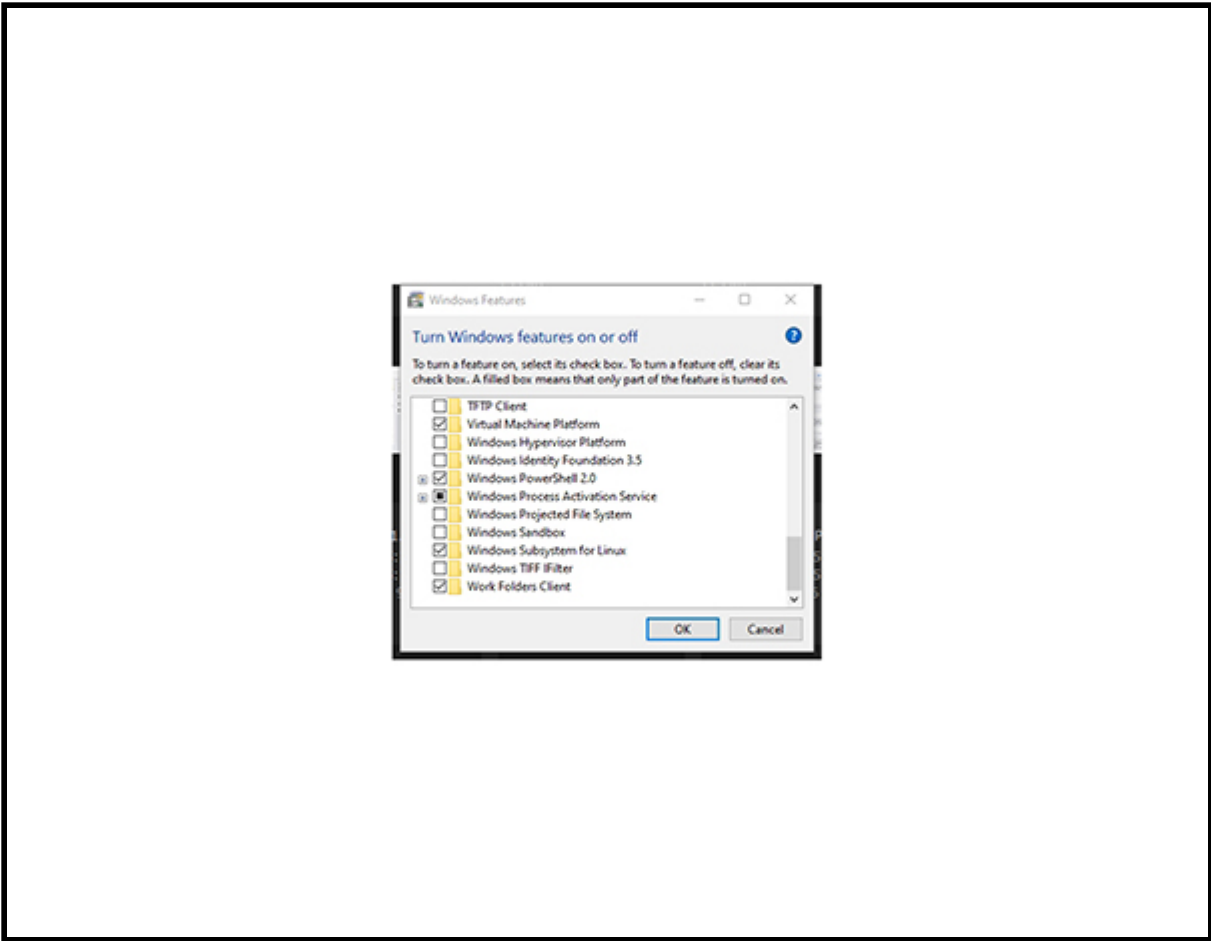
WSLInstallSequence600_0001_Layer0.jpg



WSLInstallSequence600_0002_Layer1.jpg

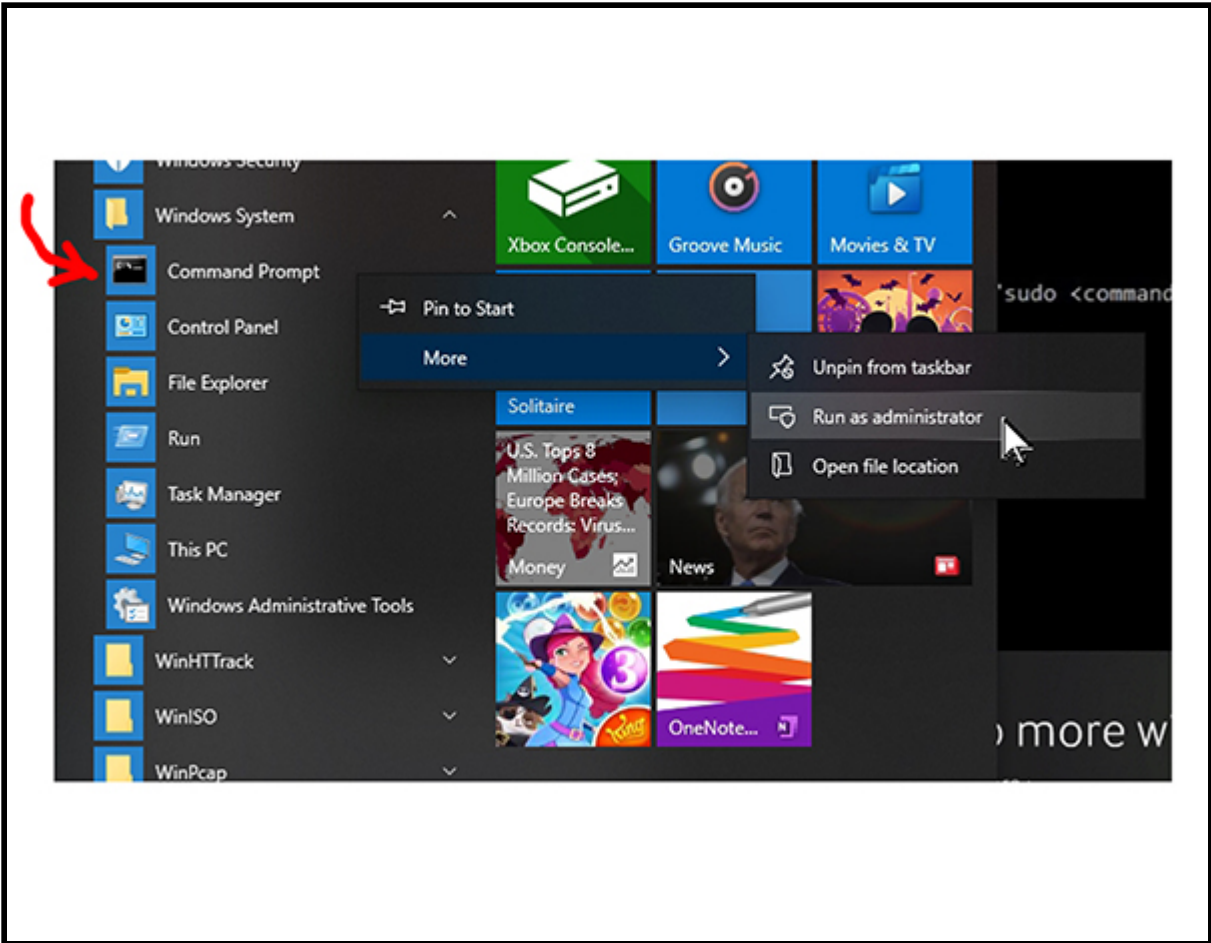


WSLInstallSequence600_0003_Layer3.jpg



WSLInstallSequence600_0004_Layer4.jpg

Next we want to tell **WSL** to run as version 2. This version has an actual **Linux** kernel instead of converting **Linux** requests to **Windows** function calls. We accomplish this through a **Windows Command Prompt** using **Run As Administrator** as shown here.



InstallHHVMSequence600_0001_Layer2.jpg

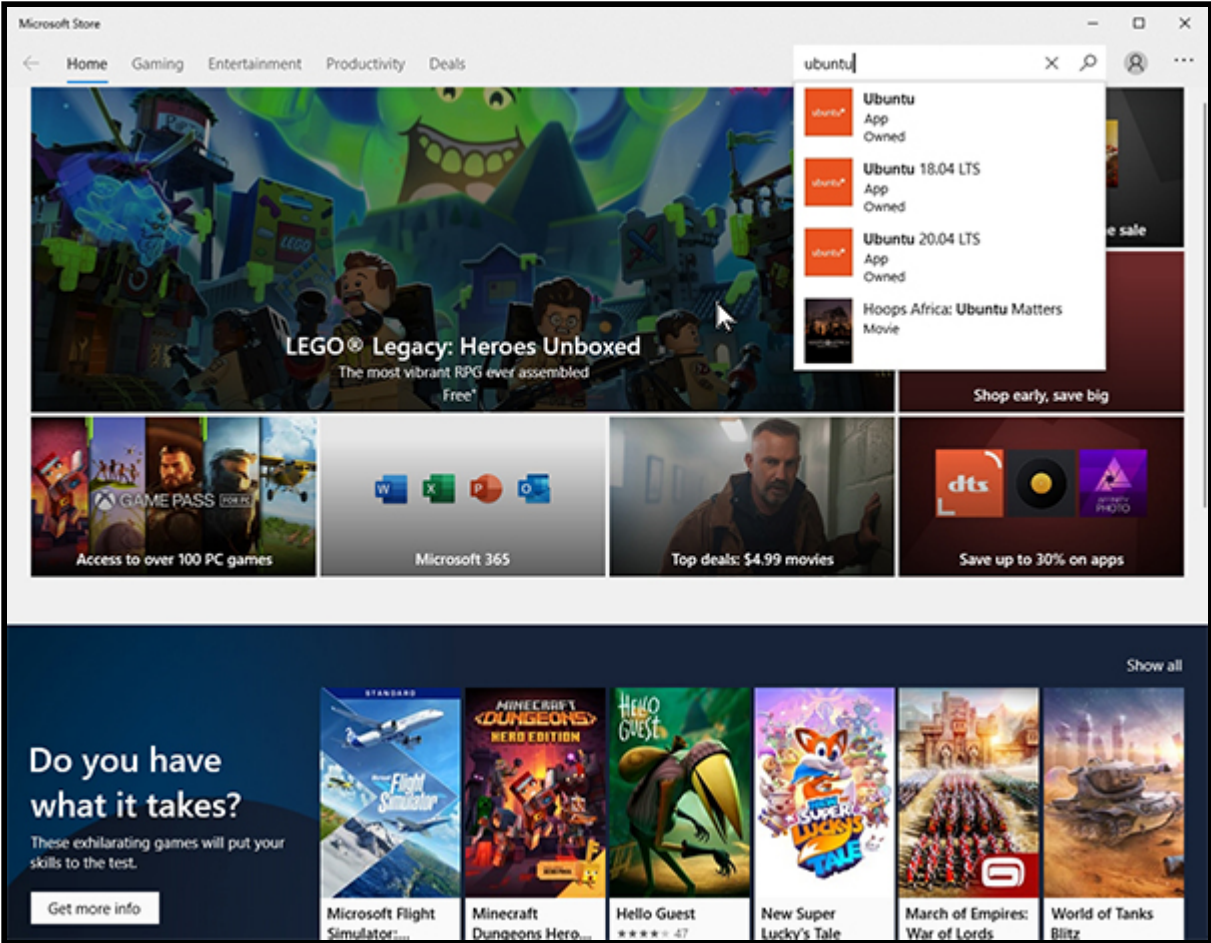
At the prompt enter the following command:

wsl --set-default-version 2

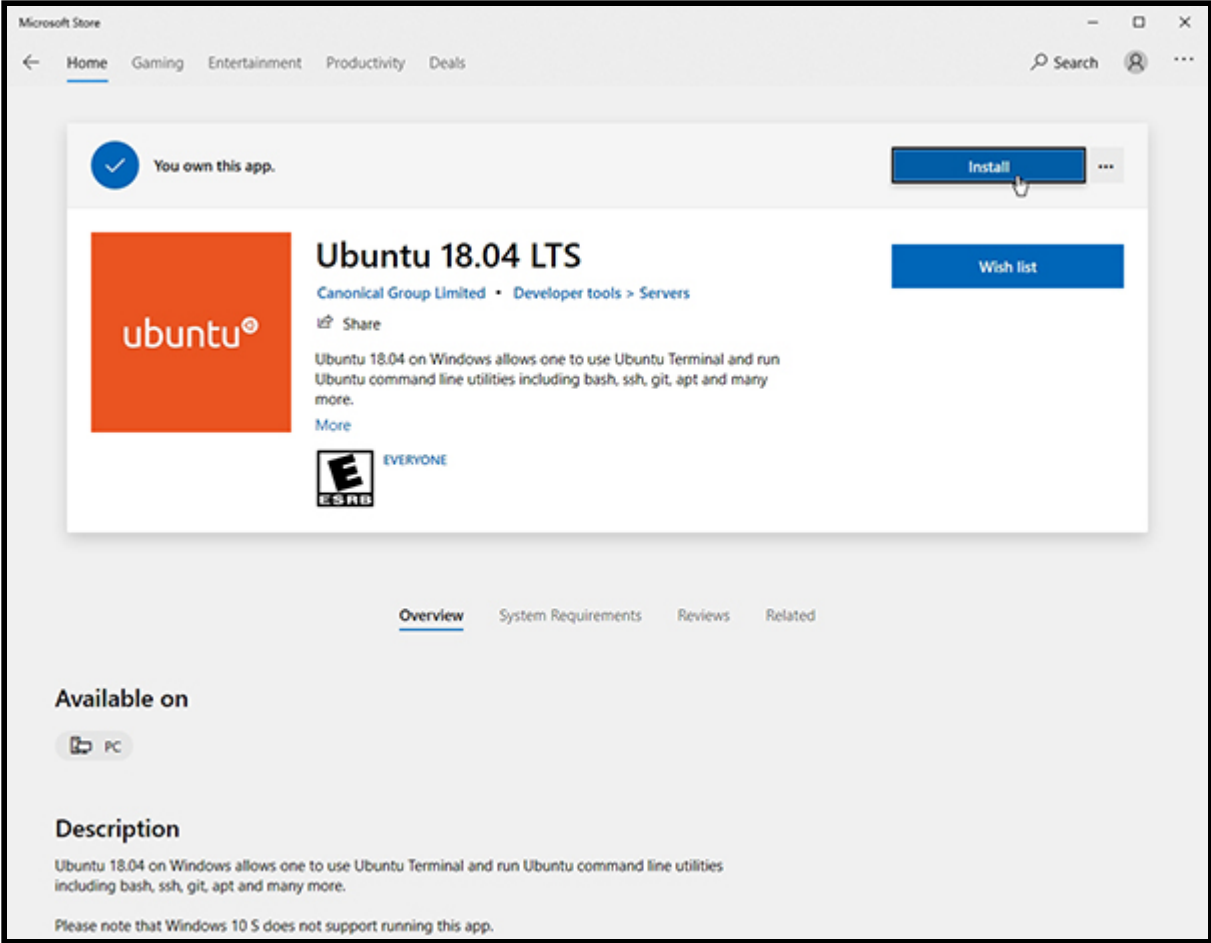
We will use the command prompt again, so leave the **Command Prompt** window open. We suggest pinning the Cmd prompt to the Taskbar.

Now we have to select and install a **Linux** distro to run under the **WSL** we just set up. Open the **Microsoft Store** and search for **Ubuntu**. In the dropdown choose **Ubuntu 18.04 LTS** from the selections and on the resulting product panel click the **Install** button at the upper right. After the distro has installed, the **Install** button will change to a **Launch** button. A launch button will also be added to the **Start Menu**. Besides those two

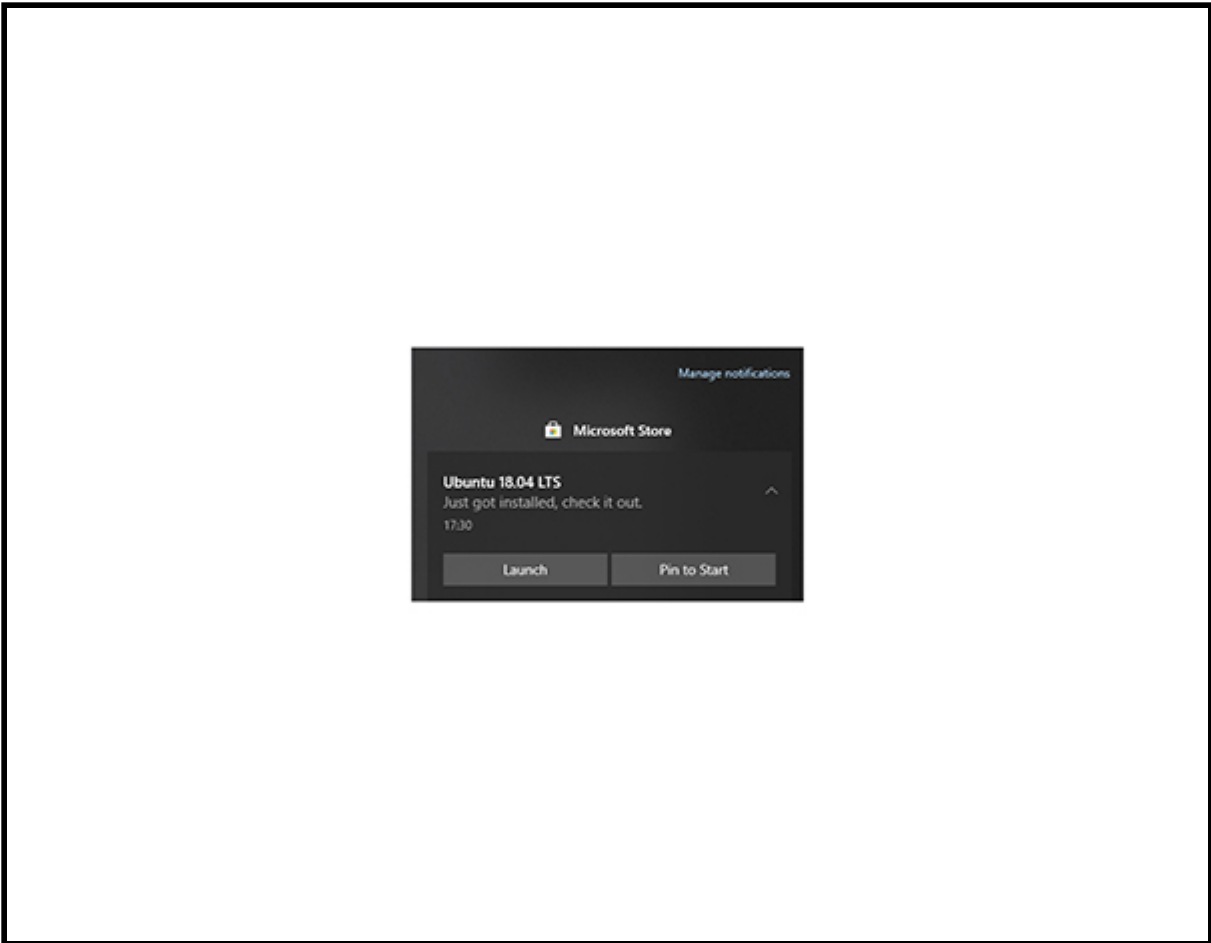
indications, **Windows** will show a message offering to launch the distro or pin it to the start menu. This activity is recapped in these screens



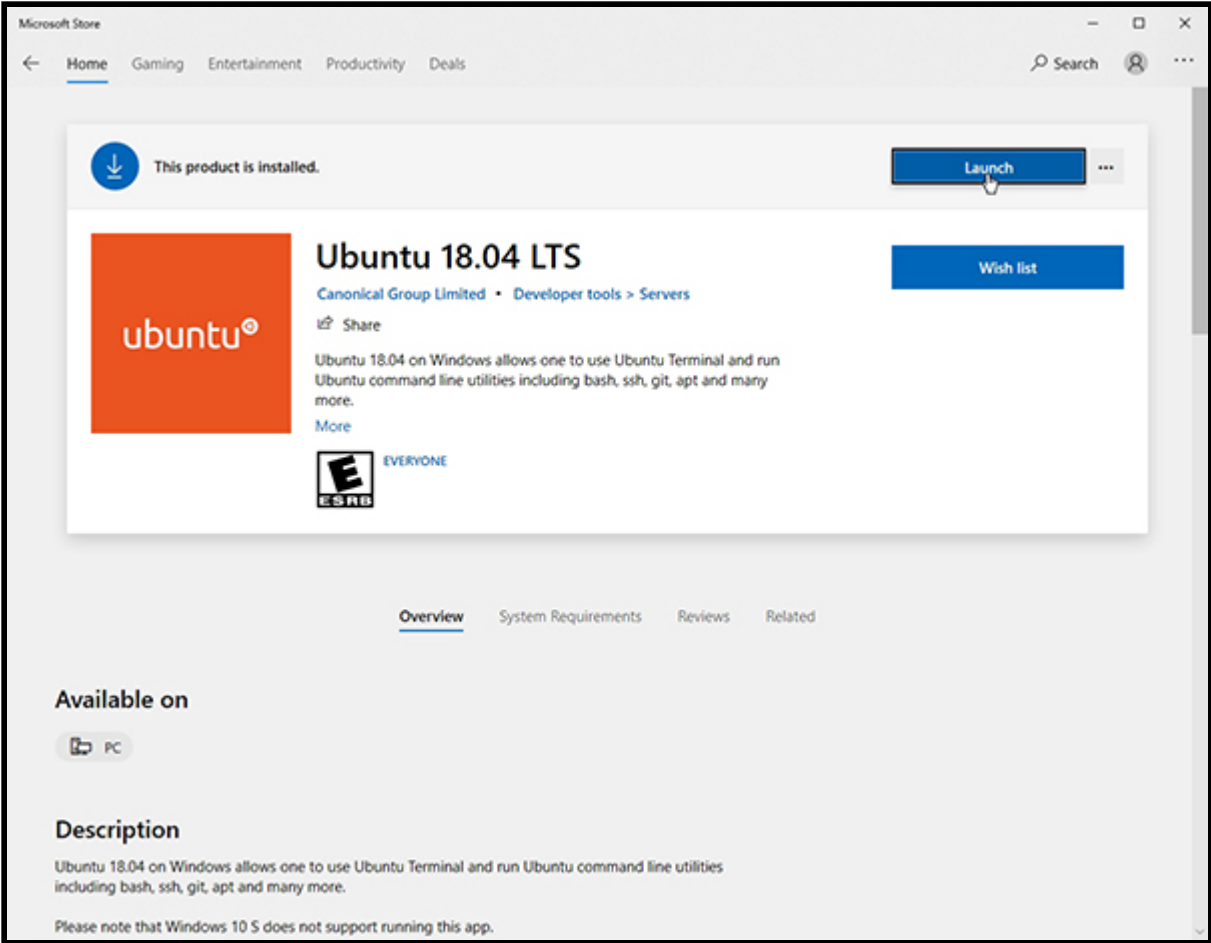
UbuntuInstallSequence600_0000_Layer0.jpg



UbuntuInstallSequence600_0001_Layer1.jpg



UbuntuInstallSequence600_0002_Layer3.jpg



UbuntuInstallSequence600_0003_Layer4.jpg

Let's just close the **Microsoft Store** window and instead click the **Ubuntu** launch button in the **Start Menu**.
In a few moments a new terminal window as shown here will open and display a message **"Installing, this may take a few minutes..."** followed by a prompt to enter a new UNIX username.

```
hhvmuser@BobosRevenge: ~$
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to match y
our windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: hhvmuser
Enter new UNIX password: enter the password Hacker
Retype new UNIX password:
passwd: password updated successfully
Installation successful!
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

hhvmuser@BobosRevenge:~$
```

UbuntuInstallSequence600_0004_enterthepasswordHacker.jpg

For our tutorial we chose the creative username of **hhvmuser**. After providing a username the system prompts for a password which will have to be entered twice for safety. Because we will probably need to provide this password during the course of operating the system we choose the short memorable password of **Hacker**. Congratulations, you have just installed **Ubuntu 18.04 LTS Linux** on your **Windows 10** machine! Pin the distro to your **Taskbar** and enter **exit** at the **Linux** prompt.

In the next sequence of operations we will do some admin housekeeping on our new distro using the **Windows Command prompt** and the **WSL** program. When **Windows** installs the **WSL** system and **Linux** distros, it places the the WSL loadable distros in the **Program Files\WindowsApps** folder. Because of drive size we may want to place our distro on another drive, but in any case we believe it is to be preferred to have our distro install in its own folder. It is also a good idea to have a sort of a backup of our installed setup. Coincidentally, moving the install to another folder creates a **tarball (TAR file)** of the installed distro.

This **Windows Command** window

```
Administrator: Command Prompt
C:\WINDOWS\system32>wsl --list --all
Windows Subsystem for Linux Distributions:
Ubuntu-18.04 (Default)

C:\WINDOWS\system32>wsl --export Ubuntu-18.04 D:\UbuntuHHVM.tar
                                     ↖ show and save our install

C:\WINDOWS\system32>wsl --unregister Ubuntu-18.04
Unregistering...
                                     ↖ remove our install

C:\WINDOWS\system32>wsl --list --all
Windows Subsystem for Linux has no installed distributions.
Distributions can be installed by visiting the Microsoft Store:
https://aka.ms/wslstore

C:\WINDOWS\system32>wsl --import Ubuntu-18.04 D:\UbuntuHHVM d:\UbuntuHHVM.tar
                                     ↖ reload our install at its
                                     new location

C:\WINDOWS\system32>
```

InstallHHVMSequence600_0014_Layer3.jpg

displays the currently installed **Linux** distros as shown here:

```
C:\WINDOWS\system32>wsl --list --all
```

Windows Subsystem for Linux Distributions:

Ubuntu-18.04 (Default)

Then enter this next command which will create a tarball archive of the specified distro as shown here:

```
C:\WINDOWS\system32>wsl --export Ubuntu-18.04 D:\UbuntuHHVM.tar
```

Now we want to remove the present distro from the WSL system and replace it with our tarballed system on another drive. We can accomplish that action with the next two commands:

```
C:\WINDOWS\system32>wsl --unregister Ubuntu-18.04
```

Unregistering...

```
C:\WINDOWS\system32>wsl --import Ubuntu-18.04 D:\UbuntuHHVM d:\UbuntuHHVM.tar
```

Now if we click our **Ubuntu** launch button on the Taskbar our **Linux** command console will reopen except that it will be using the files we just relocated. And notice that the command prompt now reflects **root** instead of **hhvmuser** as the logged on user! Our **hhvmuser** is still available as we can see with a **getent passwd** command. If we enter the command **su hhvmuser** we will switch to our ordinary user account installed previously while keeping the **root** username available by an exit command. At this point we have a **WSL** instance of **Ubuntu 18.04 LTS Linux** on our **Windows 10** system running from our computer's **D:** drive.

Let's now get to the main task of the setup, which is to update our installed distro, and then leverage that setup to download and install a **Bitnami Ubuntu** stack including the **HHVM** machine, and then to finally update that setup and save a copy as a tarball.

From our root account in the **Linux** command console run the command **apt -update** followed by the command **apt-get upgrade**. These two commands will update and upgrade the installed **Linux** system components to the latest stable versions and may take some time depending on network bandwidth and the target computer.

A link URL we can use to browse the **Bitnami LAMH stack installer** download page is

<https://bitnami.com/download/files/stacks/hhvm/3.30.12-5/bitnami-hhvm-3.30.12-5-linux-x64-installer.run> but the actual file URL we will download from is <https://bitnami.com/redirect/to/1172073/bitnami-hhvm-3.30.12-5-linux-x64-installer.run?bypassauth=false&fb=1> which we can discern by right clicking the click here link on the browser page.

In our **Ubuntu console window** change the current directory to the /opt directory with the command **cd /opt**, and then to download the installer execute the command

```
wget https://bitnami.com/redirect/to/1172073/bitnami-hhvm-3.30.12-5-linux-x64-installer.run
```

When the installer has completed its download we can observe the changes in our /opt folder:

```
root@BobosRevenge:/opt# ll
```

```
total 198592
```

```
drwxr-xr-x 2 root root 4096 Oct 16 20:28 ./
```

```
drwxr-xr-x 23 root root 4096 Oct 16 20:38 ../
```

```
-rw-r--r-- 1 root root 203348784 Oct 3 12:19 bitnami-hhvm-3.30.12-5-linux-x64-installer.run
```

To my mind one of the great beauties of **Linux** is the notion of security and self protection that it embodies in its implementation. So even though we are the root user if we now try to execute or run the installer file we will get a Permission denied error:

```
root@BobosRevenge:/opt# /opt/bitnami-hhvm-3.30.12-5-linux-x64-installer.run
```

```
-bash: /opt/bitnami-hhvm-3.30.12-5-linux-x64-installer.run: Permission denied
```

We will adjust the permissions on the installer file to allow us to run it with a **chmod** command and check our results as follows:

```
root@BobosRevenge:/opt# chmod 755 /opt/bitnami-hhvm-3.30.12-5-linux-x64-installer.run
```

```
root@BobosRevenge:/opt# ll
```

```
total 198592
```

```
drwxr-xr-x 2 root root 4096 Oct 16 20:28 ./
```

```
drwxr-xr-x 23 root root 4096 Oct 16 20:38 ../
```

```
-rwxr-xr-x 1 root root 203348784 Oct 3 12:19 bitnami-hhvm-3.30.12-5-linux-x64-installer.run*
```

We have successfully given execute permissions to the installer file, so let's run it and load our **HHVM** stack.

Following next is a screen capture transcription of the installer execution process. In each case where the installer has requested our response we just hit Enter to select the default value. For the **mySQL password** we use what we chose earlier for our **hhvmuser** account, **Hacker**. Finally, some of the displayed blank lines were omitted here.

```
root@BobosRevenge:/opt# /opt/bitnami-hhvm-3.30.12-5-linux-x64-installer.run
```

Welcome to the Bitnami HHVM Stack Setup Wizard.

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

HHVM Component : Y (Cannot be edited)

Varnish [Y/n] :

PhpMyAdmin : Y (Cannot be edited)

Is the selection above correct? [Y/n]:

Installation folder
Please, choose a folder to install Bitnami HHVM Stack
Select a folder [/opt/hhvm-3.30.12-5]:

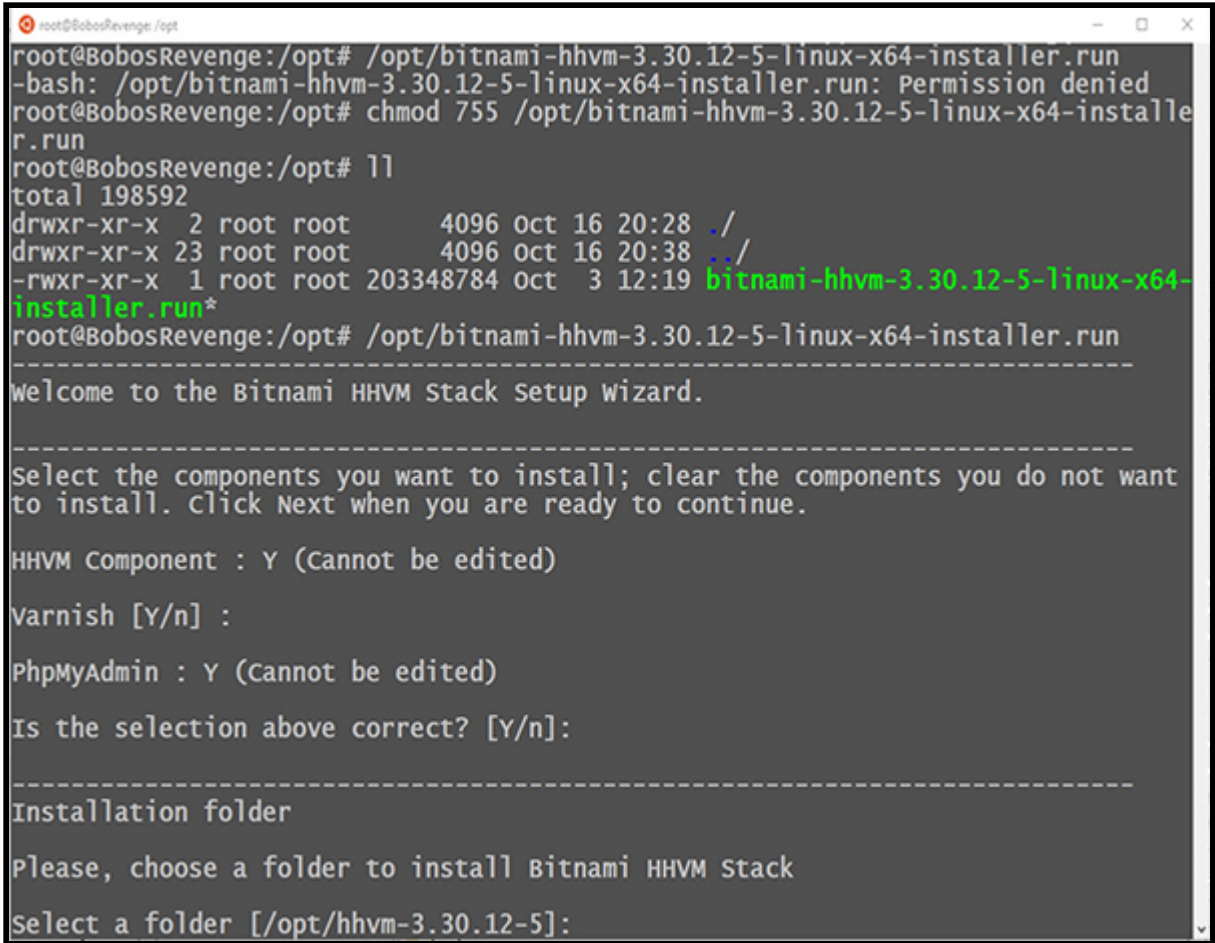
Create MySQL 'root' Account
Bitnami HHVM Stack database root user creation
Password : < we enter Hacker in response to these two requests >
Re-enter :

Setup is now ready to begin installing Bitnami HHVM Stack on your computer.
Do you want to continue? [Y/n]:

Please wait while Setup installs Bitnami HHVM Stack on your computer.
Installing
0% _____ 50% _____ 100%

Setup has finished installing Bitnami HHVM Stack on your computer.
Launch Bitnami HHVM Stack [Y/n]: y

This screen recaps some of the preceding installation activity.



InstallHHVMSequence600_0000_Layer1.jpg

We now have a Bitnami HHVM stack running in WSL on Windows 10! If we look at our setup folders we can see that a new folder **hhvm-3.30.12-5/** has been added to the **/opt** folder, and that is where our stack lives.

```
root@BobosRevenge:/opt# ll
total 198596
drwxr-xr-x 3 root root 4096 Oct 16 20:49 ./
drwxr-xr-x 23 root root 4096 Oct 16 20:38 ../
-rwxr-xr-x 1 root root 203348784 Oct 3 12:19 bitnami-hhvm-3.30.12-5-linux-x64-installer.run*
drwxr-xr-x 20 root root 4096 Oct 16 20:52 hhvm-3.30.12-5/
```

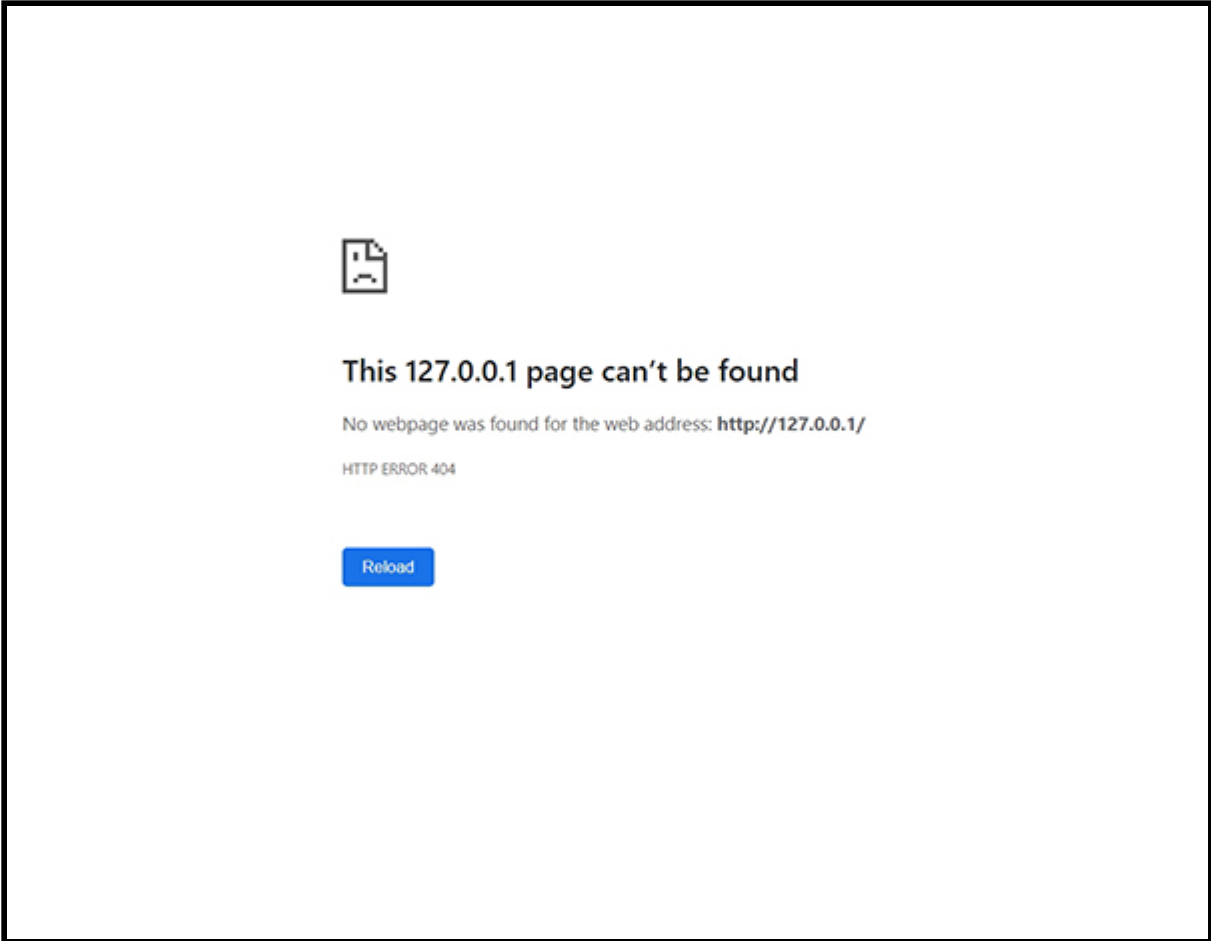
The Bitnami stack uses a script file to manage its services. Here we have the description of its usage:

```
root@BobosRevenge:/opt# /opt/hhvm-3.30.12-5/ctlscript.sh help
usage: /opt/hhvm-3.30.12-5/ctlscript.sh help
/opt/hhvm-3.30.12-5/ctlscript.sh (start|stop|restart|status)
/opt/hhvm-3.30.12-5/ctlscript.sh (start|stop|restart|status) mysql
/opt/hhvm-3.30.12-5/ctlscript.sh (start|stop|restart|status) hhvm
/opt/hhvm-3.30.12-5/ctlscript.sh (start|stop|restart|status) apache
help - this screen
start - start the service(s)
stop - stop the service(s)
```


restart - restart or start the service(s)
status - show the status of the service(s)

```
Now lets use the script to check our HHVM stack components status:  
root@BobosRevenge:/opt# /opt/hhvm-3.30.12-5/ctlscript.sh status  
hhvm already running  
apache already running  
mysql already running  
root@BobosRevenge:/opt#
```

That looks good. However, as always, looks can be deceiving. If we open a browser, say **Chrome**, on our **Windows** desktop and browse to **http://127.0.0.1** or **http://127.0.0.1/index.html** we will receive an error message that the link can't be found.



InstallHHVMSequence600_0003_Layer4.jpg

In these next two steps we will correct the issues that cause this problem. Making these corrections or modifications uses a text editor, the **Vim** editor is readily available here on our installed stack. So let's stop the stack services with a **/opt/hhvm-3.30.12-5/ctlscript.sh stop** command and make our edits to two of the configuration files.

```
root@BobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf# ll bitnami
total 20
drwxr-xr-x 2 root root 4096 Oct 16 20:51 ./
drwxr-xr-x 5 root root 4096 Oct 17 12:52 ../
-rw-r--r-- 1 root root 138 Oct 16 20:51 bitnami-apps-prefix.conf
-rw-r--r-- 1 root root 51 Oct 16 20:51 bitnami-apps-vhosts.conf
-rw-r--r-- 1 root root 2115 Oct 16 20:51 bitnami.conf
root@BobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf# vim bitnami/bitnami.conf
root@BobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf# vim httpd.conf
root@BobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf# /opt/hhvm-3.30.12-5/ctlscript.sh start
/opt/hhvm-3.30.12-5/mysql/scripts/ctl.sh : mysql started at port 3306
/opt/hhvm-3.30.12-5/hhvm/scripts/ctl.sh : hhvm started
Syntax OK
/opt/hhvm-3.30.12-5/apache2/scripts/ctl.sh : httpd started at port 80
root@BobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf# _
```

to complete our install we will edit these two files with Vim

and start the LAMH stack

InstallHHVMSequence600_0004_Layer12.jpg

One of the two issues that cause the difficulty we see are the **TCP/IP Ports** that **Apache** uses by default, which are **Port 80** and **Port 443**, the two **HTTP/HTTPS** ports ordinarily assumed for those connections. By default ports lower than 1024 can only be used by the root user. So we will modify those two port values in two configuration files using Vim. Both of the files are under the same folder so let's go there with a **cd /opt/hhvm-3.30.12-5/Apache2/conf** command. An **ll** command will show a subfolder, **bitnami**, and that folder contains a file named **bitnami.conf** which contains some of the Apache directives relative to Port usage. Let's edit that file with the command **vim bitnami/bitnami.conf** to make these changes. Good practice would suggest backing up the files before editing.

```
# Default Virtual Host configuration.

<IfVersion < 2.3 >
    NameVirtualHost *:80
    NameVirtualHost *:443
</IfVersion>

<VirtualHost _default_:80>
    DocumentRoot "/opt/hhvm-3.30.12-5/apache2/htdocs"
    <Directory "/opt/hhvm-3.30.12-5/apache2/htdocs">
        Options Indexes FollowSymLinks
        AllowOverride All
        <IfVersion < 2.3 >
            Order allow,deny
            Allow from all
        </IfVersion>
        <IfVersion >= 2.3 >
            Require all granted
        </IfVersion>
    </Directory>

    # Error Documents
    ErrorDocument 503 /503.html

    # Bitnami applications installed with a prefix URL (default)
    Include "/opt/hhvm-3.30.12-5/apache2/conf/bitnami/bitnami-apps-prefix.conf"
</VirtualHost>

# Default SSL Virtual Host configuration.

<IfModule !ssl_module>
    LoadModule ssl_module modules/mod_ssl.so
</IfModule>

Listen 443
SSLProtocol all -SSLv2 -SSLv3
SSLHonorCipherOrder on
SSLCipherSuite "EECDH+ECDSA+AESGCM EECDH+aRSA+AESGCM EECDH+ECDSA+SHA384 EECDH+EC
"bitnami/bitnami.conf" 69L, 2115C
```

change the marked Port numbers

change each 80 to 8080
change each 443 to 8443

InstallHHVM Sequence600_0005_Layer6.jpg

```
root@bobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf
<IfModule !ssl_module>
  LoadModule ssl_module modules/mod_ssl.so
</IfModule>

Listen 443
SSLProtocol all -SSLv2 -SSLv3
SSLHonorCipherOrder on
SSLCipherSuite "EECDH+ECDSA+AESGCM EECDH+aRSA+AESGCM EECDH+ECDSA+SHA384 EECDH+EC
DSA+SHA256 EECDH+aRSA+SHA384 EECDH+aRSA+SHA256 EECDH !aNULL !eNULL !LOW !3DES !M
D5 !EXP !PSK !SRP !DSS !EDH !RC4"
SSLPassphraseDialog builtin
SSLSessionCache "shmcb:/opt/hhvm-3.30.12-5/apache2/logs/ssl_scache(512000)"
SSLSessionCacheTimeout 300

<VirtualHost _default_: 443>
  DocumentRoot "/opt/hhvm-3.30.12-5/apache2/htdocs"
  SSLEngine on
  SSLCertificateFile "/opt/hhvm-3.30.12-5/apache2/conf/server.crt"
  SSLCertificateKeyFile "/opt/hhvm-3.30.12-5/apache2/conf/server.key"

  <Directory "/opt/hhvm-3.30.12-5/apache2/htdocs">
    Options Indexes FollowSymLinks
    AllowOverride All
    <IfVersion < 2.3 >
      Order allow,deny
      Allow from all
    </IfVersion>
    <IfVersion >= 2.3 >
      Require all granted
    </IfVersion>
  </Directory>

  # Error Documents
  ErrorDocument 503 /503.html

  # Bitnami applications installed with a prefix URL (default)
  Include "/opt/hhvm-3.30.12-5/apache2/conf/bitnami/bitnami-apps-prefix.conf"
</VirtualHost>

-- INSERT --
```

InstallHHVMSequence600_0006_Layer7.jpg

Within the **bitnami.conf** configuration file near the top locate the two lines containing **NameVirtualHost** with the values of **80** and **443** respectively. Change the two values to **8080** and **8443** respectively. Below that initial section is a section for HTTP and another for HTTPS. Change the value of **VirtualHost _default_:** in the HTTP section from **80** to **8080** and in the HTTPS section change the value from **443** to **8443**. Save the modified **bitnami.conf** file.

The second configuration file we will modify to change the default Port values also contains directives that we will modify to correct our second issue, the **HHVM** handler. When we complete this edit we should have a fully functional HHVM stack running on **Windows 10**. So invoke the Vim editor on the **httpd.conf** file in our **conf** folder. The port values we want to change are contained in two lines within the file. The first line is **Listen 80**, and we want to change that to **Listen 127.0.0.1:8080** and the second line is **ServerName localhost:80** and we want to change that to **ServerName 127.0.0.1:8080**.

```
root@bobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf
# running httpd, as with most system services.
#
User daemon
Group daemon

</IfModule>

# 'Main' server configuration
#
# The directives in this section set up the values used by the 'main'
# server, which responds to any requests that aren't handled by a
# <VirtualHost> definition. These values also provide defaults for
# any <VirtualHost> containers you may define later in the file.
#
# All of these directives may appear inside <VirtualHost> containers,
# in which case these default settings will be overridden for the
# virtual host being defined.
#
#
# ServerAdmin: Your address, where problems with the server should be
# e-mailed. This address appears on some server-generated pages, such
# as error documents. e.g. admin@your-domain.com
#
ServerAdmin you@example.com
#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If your host doesn't have a registered DNS name, enter its IP address here.
#
ServerName localhost:80
#
# Deny access to the entirety of your server's filesystem. You must
# explicitly permit access to web content directories in other
# <Directory> blocks below.
```

InstallHHVMSequence600_0008_Layer8.jpg


```
root@bobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf
# will be interpreted as '/logs/access_log'.
#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# Do not add a slash at the end of the directory path. If you point
# ServerRoot at a non-local disk, be sure to specify a local disk on the
# Mutex directive, if file-based mutexes are used. If you wish to share the
# same ServerRoot for multiple httpd daemons, you will need to change at
# least PidFile.
#
ServerRoot "/opt/hhvm-3.30.12-5/apache2"
#
# Mutex: Allows you to set the mutex mechanism and mutex file directory
# for individual mutexes, or change the global defaults
#
# Uncomment and change the directory if mutexes are file-based and the default
# mutex file directory is not on a local disk or is not appropriate for some
# other reason.
#
# Mutex default:logs
#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
Listen 12.34.56.78:80
Listen 80 ← replace this with → Listen 127.0.0.1:8080
#
# Dynamic Shared Object (DSO) support
#
# To be able to use the functionality of a module which was built as a DSO you
```

InstallHHVMSequence600_0007_Layer9.jpg

Scan for the line **IfModule !php7_module** and the next few lines following.

```
root@bobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf
# Note: The following must be present to support
# starting without SSL on platforms with no /dev/random equivalent
# but a statically compiled-in mod_ssl.
#
<IfModule ssl_module>
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
</IfModule>

Include "conf/deflate.conf"
#Include conf/pagespeed.conf
#Include conf/pagespeed_libraries.conf

AddType application/x-httpd-php .php .phpml

# This enables using PHP-FPM when mod_php is disabled
<IfModule !php7_module>
    Action application/x-httpd-php "/bitnami-error-php-fpm-did-not-handle-the-co
nnection"
    Define USE_PHP_FPM
    Include "conf/php-fpm-apache.conf"
</IfModule>

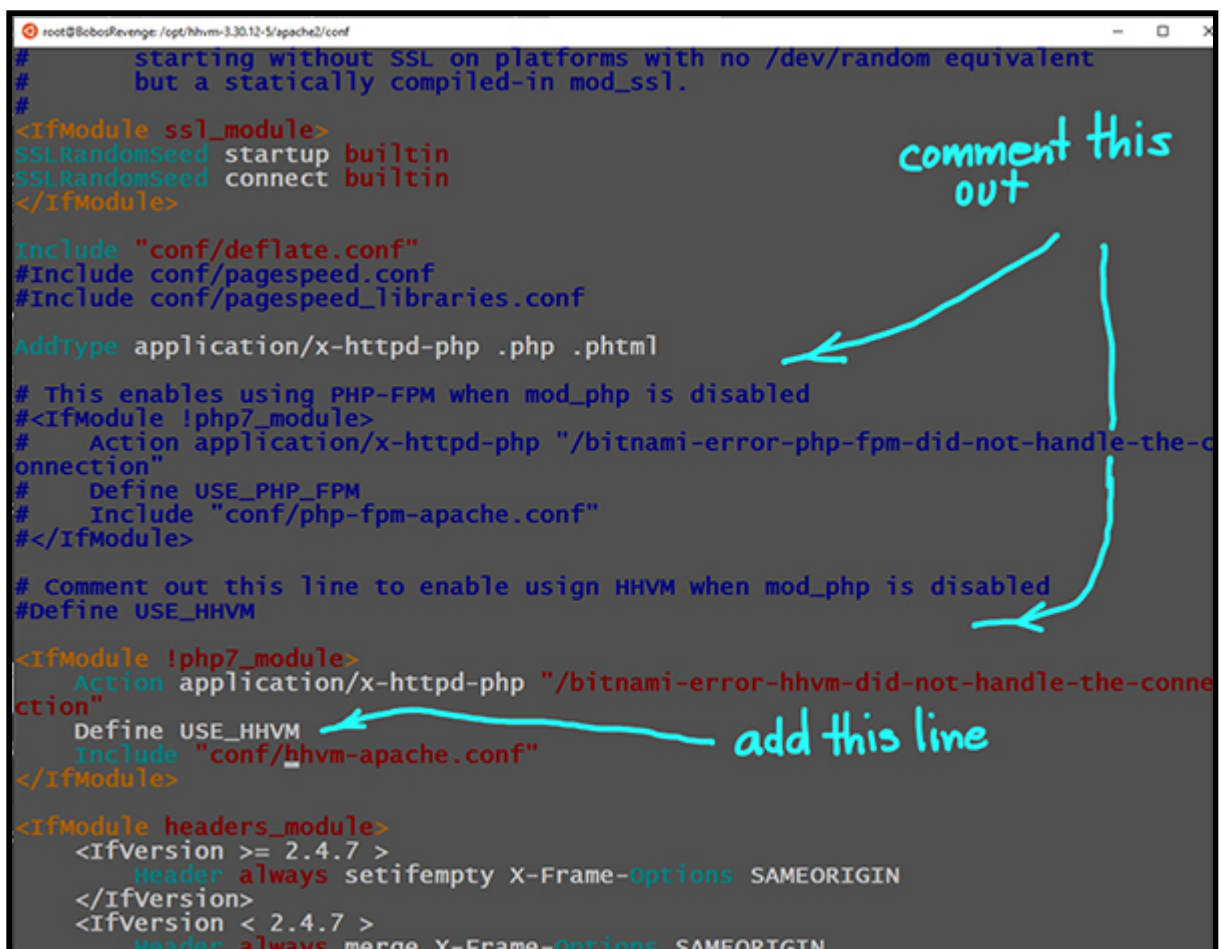
# Comment out this line to enable usign HHVM when mod_php is disabled
Define USE_HHVM

<IfModule !php7_module>
    Action application/x-httpd-php "/bitnami-error-hhvm-did-not-handle-the-conne
ction"
    Include "conf/hhvm-apache.conf"
</IfModule>

<IfModule headers_module>
    <IfVersion >= 2.4.7 >
        header always setifempty X-Frame-Options SAMEORIGIN
    </IfVersion>
    <IfVersion < 2.4.7 >
        header always merge X-Frame-Options SAMEORIGIN
```

InstallHHVMSequence600_0009_Layer10.jpg

The first such declaration will contain a line with **Define USE_PHP_FPM** and we want to comment that declaration and its contents out or delete it. We can comment each line by using the hash or pound sign # at the beginning of each line. Now below the section we have just disabled is a second **IfModule !php7_module** declaration, and just before that section appears is a line **Define USE_HHVM**. We want to comment that **Define USE_HHVM** line out and add this statement within the **IfModule !php7_module** section just below it. Insert this line between the **Action** and **Include** lines so the syntax matches our commented out section above. We want the end results of this set of edits to look like this:



```
root@BobosRevenge: /opt/hhvm-3.30.12-5/apache2/conf
#
# starting without SSL on platforms with no /dev/random equivalent
# but a statically compiled-in mod_ssl.
#
<IfModule ssl_module>
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
</IfModule>

Include "conf/deflate.conf"
#Include conf/pagespeed.conf
#Include conf/pagespeed_libraries.conf

AddType application/x-httpd-php .php .phtml

# This enables using PHP-FPM when mod_php is disabled
#<IfModule !php7_module>
# Action application/x-httpd-php "/bitnami-error-php-fpm-did-not-handle-the-c
onnection"
# Define USE_PHP_FPM
# Include "conf/php-fpm-apache.conf"
#</IfModule>

# Comment out this line to enable usign HHVM when mod_php is disabled
#Define USE_HHVM

<IfModule !php7_module>
Action application/x-httpd-php "/bitnami-error-hhvm-did-not-handle-the-conne
ction"
Define USE_HHVM
Include "conf/hhvm-apache.conf"
</IfModule>

<IfModule headers_module>
<IfVersion >= 2.4.7 >
Header always setifempty X-Frame-Options SAMEORIGIN
</IfVersion>
<IfVersion < 2.4.7 >
Header always merge X-Frame-Options SAMEORIGIN
</IfVersion>
</IfModule>
```

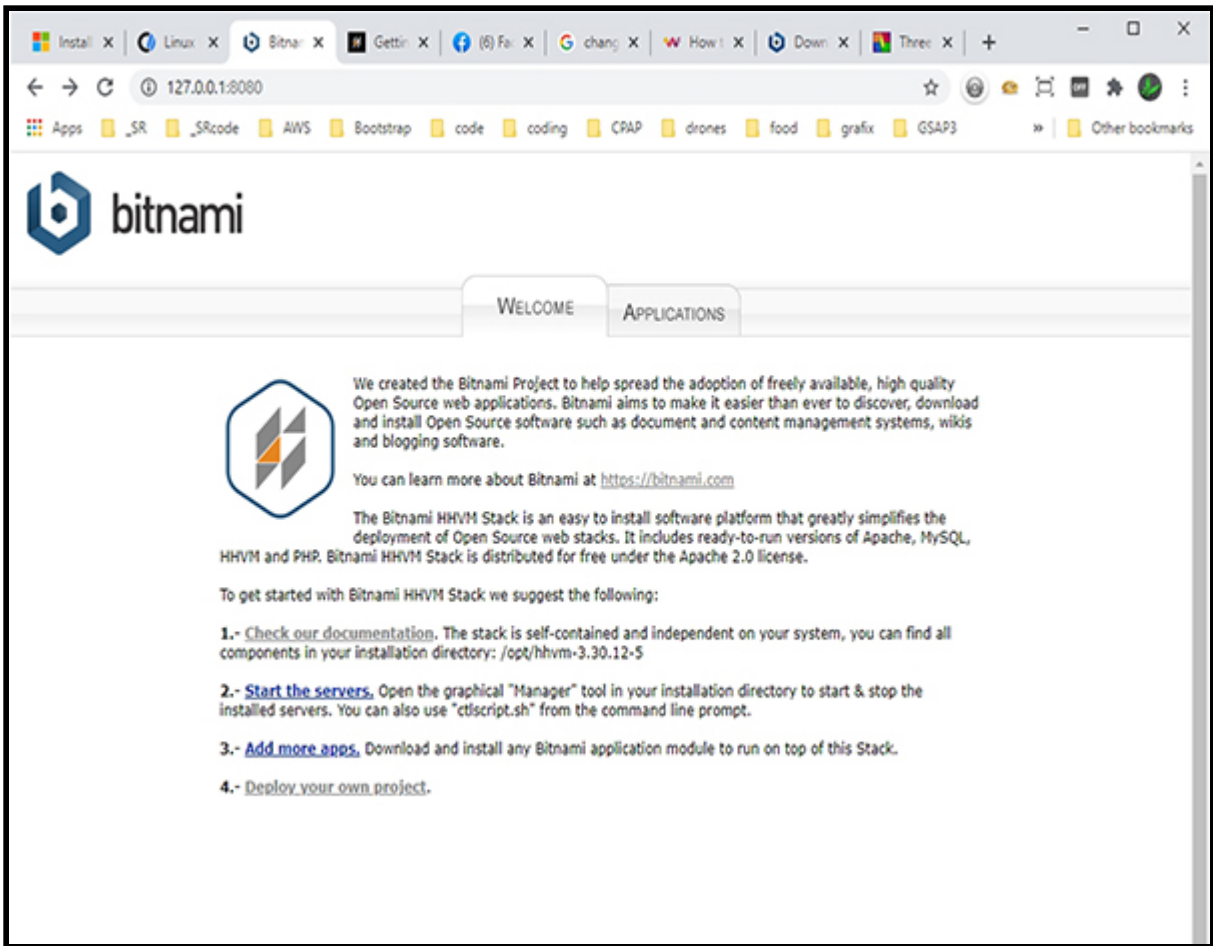
InstallHHVMSequence600_0010_Layer11.jpg

Note the less than and greater than delimiters are replaced with [lt] and [gt].

```
# This enables using PHP-FPM when mod_php is disabled
#[lt]IfModule !php7_module[gt]
# Action application/x-httpd-php "/bitnami-error-php-fpm-did-not-handle-the-connection"
# Define USE_PHP_FPM
# Include "conf/php-fpm-apache.conf"
#[lt]/IfModule[gt]
# Comment out this line to enable usign HHVM when mod_php is disabled
#Define USE_HHVM
[lt]IfModule !php7_module[gt]
Action application/x-httpd-php "/bitnami-error-hhvm-did-not-handle-the-connection"
Define USE_HHVM
Include "conf/hhvm-apache.conf"
[lt]/IfModule>[gt]
```

When these edits are done save the modified file.

Time for a quick reward! In our Ubuntu console execute the command to start our HHVM stack services
root@BobosRevenge:/opt# /opt/hhvm-3.30.12-5/ctlscript.sh start Now open your **Windows** browser and go to the URL **http://127.0.0.1:8080** where you should be greeted by the default **Bitnami HTML** home page.



InstallHHVMSequence600_0011_Layer13.jpg

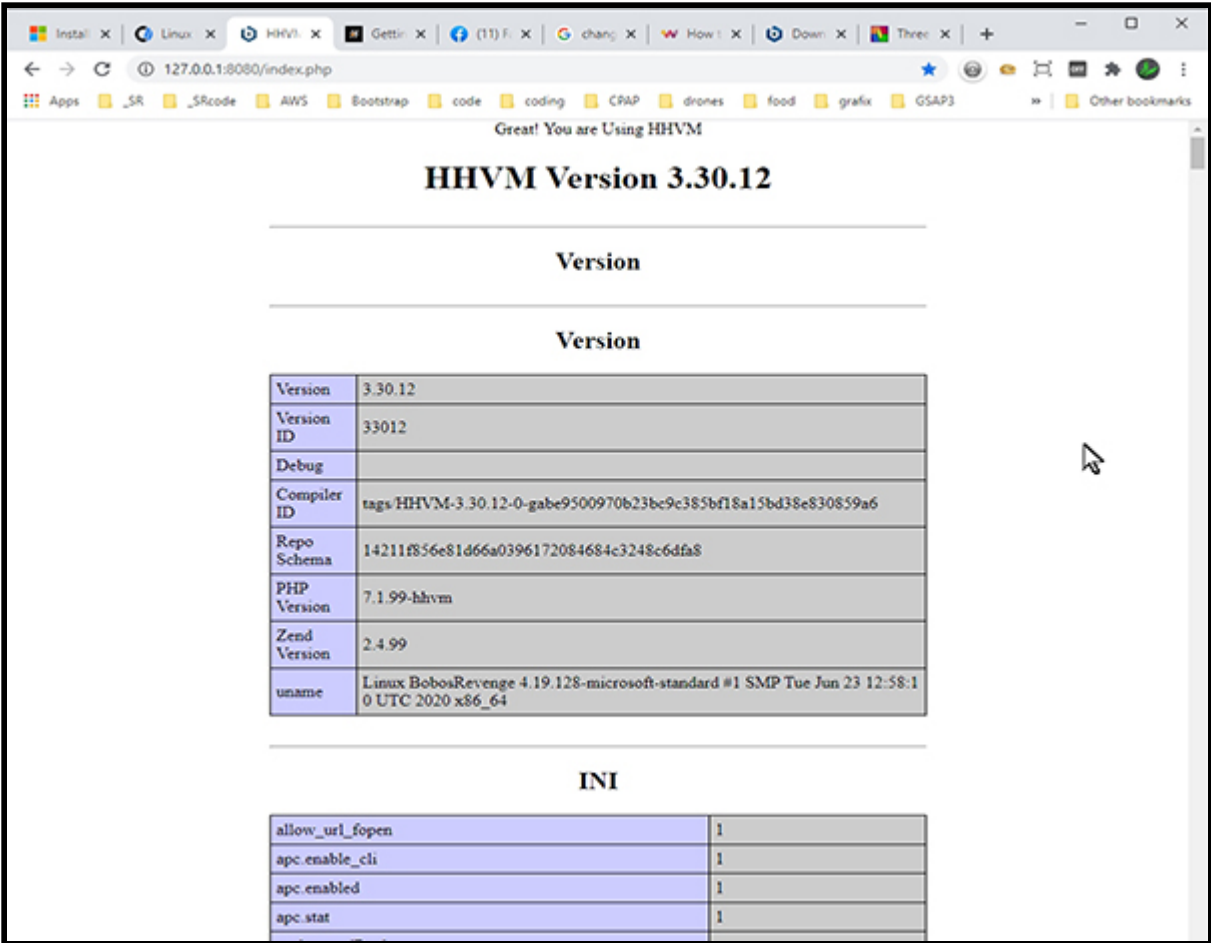
Finally let's test the **HHVM** machine. Navigate to the **htdocs** folder where our web content is stored with a **cd /opt/hhvm-3.30.12-5/apache2/htdocs** command and invoke the Vim editor to create a new file, **index.php** within that folder. Insert the following lines into the file and save it, and exit the editor. Note the less than and greater than delimiters are replaced with [lt] and [gt].

```
[lt]?php
echo defined('HHVM_VERSION') ? 'Great! You are Using HHVM' : 'Sorry! Not using HHVM';
phpinfo();
?[gt]
```



InstallHHVMSequence600_0012_Layer0.jpg

Now return to the **Windows** desktop browser and browse to the URL **http://127.0.0.1:8080/index.php** and you should see this screen.



InstallHHVMSequence600_0013_Layer14.jpg

That completes the installation of the **HHVM stack** with the exception of creating a backup tarball archive and reloading our stack from the tarball. By doing that we always have a known base install to fall back on. The following three commands should be run in the **Windows Command prompt window** we used previously:

```
C:\WINDOWS\system32>wsl --export Ubuntu-18.04 D:\UbuntuHHVMbase.tar
C:\WINDOWS\system32>wsl --unregister Ubuntu-18.04
Unregistering...
C:\WINDOWS\system32>wsl --import Ubuntu-18.04 D:\UbuntuHHVM d:\UbuntuHHVMbase.tar
```

When these operations complete you will have an easy to reload and duplicate Ubuntu 18.04 Linux machine with an Apache, mySQL, and HHVM stack running on Windows 10 using the Windows Subsystem for Linux service.

If you have another **Windows 10** machine that has the same drive assignments available (we used the **C:** drive and the **D:** drive, so the target has to have two logical disks as well) we can simply copy our tarball to the second machine. Install the **WSL** services as we did here above through the **Settings** dialog and then, in the **Windows Command prompt** on the target machine, we can import our tarball as we did here above using the command

```
C:\WINDOWS\system32>wsl --import Ubuntu-18.04 D:\UbuntuHHVM d:\UbuntuHHVMbase.tar
```

Now if we enter this next command we see the result as shown here.

```
C:\WINDOWS\system32>wsl --list --all
```

Windows Subsystem for Linux Distributions:

Ubuntu-18.04 (Default)

In order to get this newly copied install to run open the **Windows Store** and search for **Ubuntu**, choose the **Ubuntu 18.04 LTS** version, and then click the **Install** button. When the package has downloaded and installed, the button will change to say **Launch**. Click that and it will launch the previously downloaded **LAMH** stack. Pin the button to the taskbar. Now you have a duplicate of your base install on the second machine.

I like writing code and I like learning how to write it. I want to learn HACK. This lets me do that on a local machine.

Information and Knowledge is Power. Power will get you Freedom, and Freedom is not negotiable. I have a huge debt of gratitude for all the tutorials and forums and blogs and Github sites that have helped me learn, and hopefully this tutorial will help me pay the debt forward by helping you.