WordPress blog site on Azure.

If you would like to start your own blog, are technically minded, don't mind installing free software and also want the blog web-site to be professional looking without writing much code, then you've reached the right address on the net.

This article explains provides the step-by-step guide on creating a professional blog web site using LAMP[1] stack with WordPress® on Azure.

Caveat: Although this document provides step-by-step guide as much as possible, reader is expected to have familiarity with Linux and Azure (Microsoft cloud offering). Most of the scripts needed for this setup are available from this GitHub repository: https://github.com/ninadkan/WordPressBlogOnLampAndAzure

Azure Subscription

First things first, you'll need an Azure subscription. If you don't have any, you can get a free subscription for 12 months by signing up. Navigate to https://signup.azure.com.

At the time of writing this blog, one could take advantage of £150 ($200) credit for 30 days. Whilst registering, you'll need an email, a valid phone number and a credit card for authentication and verification.

[MANUAL STEP]: Buy Azure subscription.

BUT: Your subscription will require the ability to configure purchased custom domain names. This feature is not available with the free subscription and you'll need to remove the spending limits on your subscription. Removing the spending limits will allow your App service domains running under Azure to register the custom domain names. The instructions on how to achieve this manually is available here.

The same result can be achieved using the PowerShell script available from the GitHub repo here. 'install.ps1' script first creates a resource group, next creates an app service plan. Whilst creating the app service plan, 'D1' SKU is specified which moves the subscription out of the Free Tier layer. Finally, script creates a web app. This web app will be used to establish connectivity with the purchased custom domain name.

[AUTOMATION STEP]: Execute the script 'install.ps1' from the Windows PowerShell console.

[VALIDATION TESTS]: Log on to Azure portal – portal.azure.com and manually validate that the region, app plan and web app are created.

NOTE: Automation scripts are written in Azure PowerShell. All the global parameters are specified in 'parameters.ps1' file. All scripts include common 'login.ps1' script which validates that the user is logged in and ensures that the correct subscription is selected. If you want to clean the resources, use the script 'uninstall.ps1'.

[1] LAMP → Linux operating system, Apache web server, MySQL as database and PHP as the scripting/programming language.

Purchase Domain Name

To set-up a personalised blog, a URL, for e.g. '*ninadkanthi.co.uk*', needs to be registered. To achieve this, the domain name needs to be registered with ICANN-Registered Registrars. If you've already got the domain name, you can skip the rest of this section.

There are two options – either buy one from the other providers, like 123-reg.co.uk, goDaddy.com, bluehost.com; search for domain name registration in your web browser. Or you can buy one from the Azure portal itself. The detailed explanation of the former process is explained here and for the latter here

NOTE: If you are new to DNS, Zone, Records etc, recommend that you buy the domain name within Azure portal itself. if you pursue this path, i.e. buying the app service domain from the azure portal, the domain registration is provided by GoDaddy.com, LLC ("Go Daddy"), who would be registrar of record. Cost, at the time of writing this document was $11.99 per year.

NOTE: If you buy the domain name from the Azure portal, you have five days to cancel the transaction.

[MANUAL STEP] For this exercise, I took the option of buying just the domain name – '*ninadkanthi.co.uk*' – for an year directly from the Godaddy.com website – cost was £1.20 (inclusive of tax for the first year!).

[VALIDATION TEST]: You should now be able to login to your domain registrar web site and access your DNS records.

Bind a Host Name to the web app in Azure

[AUTOMATION STEP]: Execute the script 'add-host-name.ps1' from the Windows PowerShell console. Validate that a successful result is returned.

Snippet of the script is shown below:

```
$AlreadyAdded=0
$HostNameList = az webapp config hostname list --webapp-name $SERVICEPLAN_APPNAME -g
$RESOURCEGROUP_NAME --subscription $SUBSCRIPTION --query [].name -o json

foreach($element in $HostNameList){
    $trimmedElement = TrimVariable $element

    if ($trimmedElement  -contains $FQDN)
    {
        echo "hostname $FQDN already exists!"
        $AlreadyAdded=1
        break
    }
}

if ($AlreadyAdded -eq 0)
{
    echo "Adding hostname $FQDN"
    az webapp config hostname add --webapp-name $SERVICEPLAN_APPNAME -g $RESOURCEGROUP_NAME
--subscription $SUBSCRIPTION --hostname $FQDN
}
```

[VALIDATION TEST]: Login to Azure portal (portal.azure.com). Open the Custom Hostnames blade (App Services → Select App Service → Custom Domains). Verify that the custom domain name is now mapped as a valid Custom Hostname.

[VALIDATION TEST] You should now be able to navigate to your custom domain (*ninadkanthi.co.uk*) from your browser and see the default Azure 'Your App Service app is up and running' page being displayed by Azure.

Installing and configuring the Linux server

As mentioned earlier, we'll host the WordPress site on the LAMP stack. This section covers creation and configuration of the Linux instance of the LAMP stack.

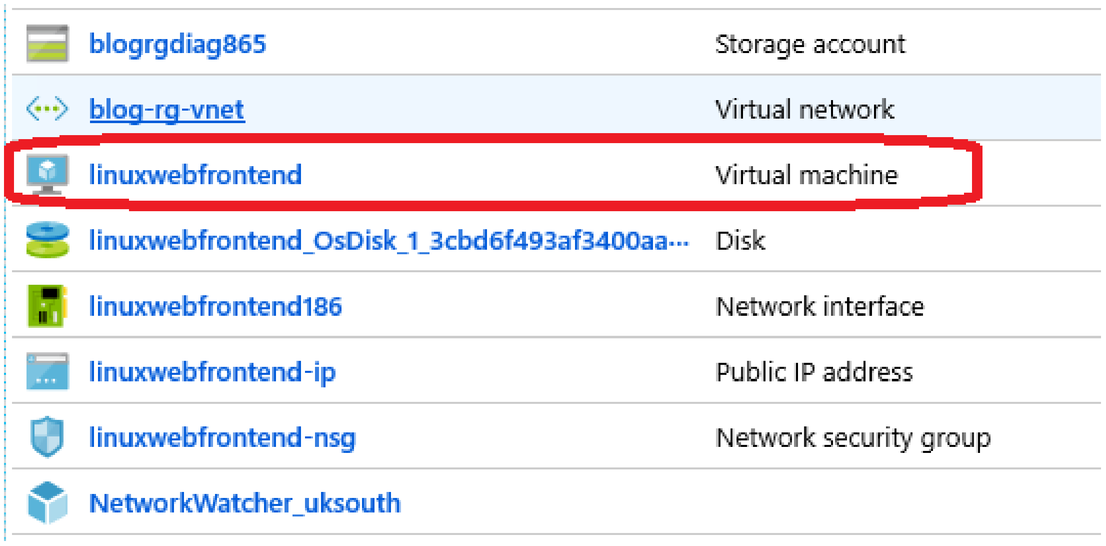
For the Linux variant, I've decided to use Ubuntu 16.04-LTS version with local SSD storage. The virtual machine size was chosen to be 'B2' as I believe it provides the best balance between cost and performance.

To install the Linux server, ARM template scripts were used and are copied here

[AUTOMATION STEP]: To install the Linux server, first edit the '**parameters.json**' file with your settings, especially need to add the '**adminPublicKey**'; by default, only SSH login is permitted to the instance.

Next execute the '**deploy.ps1**' script; will need to specify the subscription name, resource name and deployment name

[VALIDATION TEST]: Logon to the azure portal and confirm that the Linux compute had been provisioned under the chosen resource group. Besides the virtual machine, following other components and services should also be added under the resource group.



blogrgdiag865	Storage account
blog-rg-vnet	Virtual network
linuxwebfrontend	Virtual machine
linuxwebfrontend_OsDisk_1_3cbd6f493af3400aa...	Disk
linuxwebfrontend186	Network interface
linuxwebfrontend-ip	Public IP address
linuxwebfrontend-nsg	Network security group
NetworkWatcher_uksouth	

Adding Non-Root user to the Linux server

Login to the Linux web server (using Putty or any other Telnet terminal application). Before installing Apache and other software stack, we'll add another user to the Linux machine and run everything under this user. This reduces the attack surface impact in terms of security compromise.

[MANUAL STEP] Copy the script 'addconfigureuser.sh' to the Linux server (using pscp or equivalent program, or download directly from the GitHub repository onto the Linux machine). Ensure that the script is marked as an executable (execute command '**chmod + x addconfigureuser.sh**')

Script creates a user named '**webuser**'. During its execution, the script will ask for a password for the 'webuser'. Ensure that you specify a strong password and make a note of that password.

```bash
#!/bin/bash
# this  file should be copied to the root of remote user home folder
# chmod +x on this and then executed.
# this will add a webuser which will be used to install all the local
# software
sudo adduser webuser
sudo usermod -aG sudo webuser
su - webuser
```

[VALIDATION TEST]: From the terminal window, execute command **id -u webuser**. If it returns a +ve number, it indicates that the user creation was successful.

[VALIDATION TEST]: After successful execution of script, you should now be logged in as the newly created user – **webuser**.

Install and configure Apache

[Ensure that you're logged in as '**webuser'** on the Linux machine; if not execute command **su - webuser** from the terminal prompt]

[MANUAL STEP] Copy scripts under the 'webuser' of the GitHub repository to the Linux server (using pscp, direct from GitHub, or any other equivalent technique). Mark each script as an executable (hint use **chmod +x** …)

[AUTOMATION STEP]: First check that the user has the permissions to install software. Execute the script 'checkandupdate.sh'.

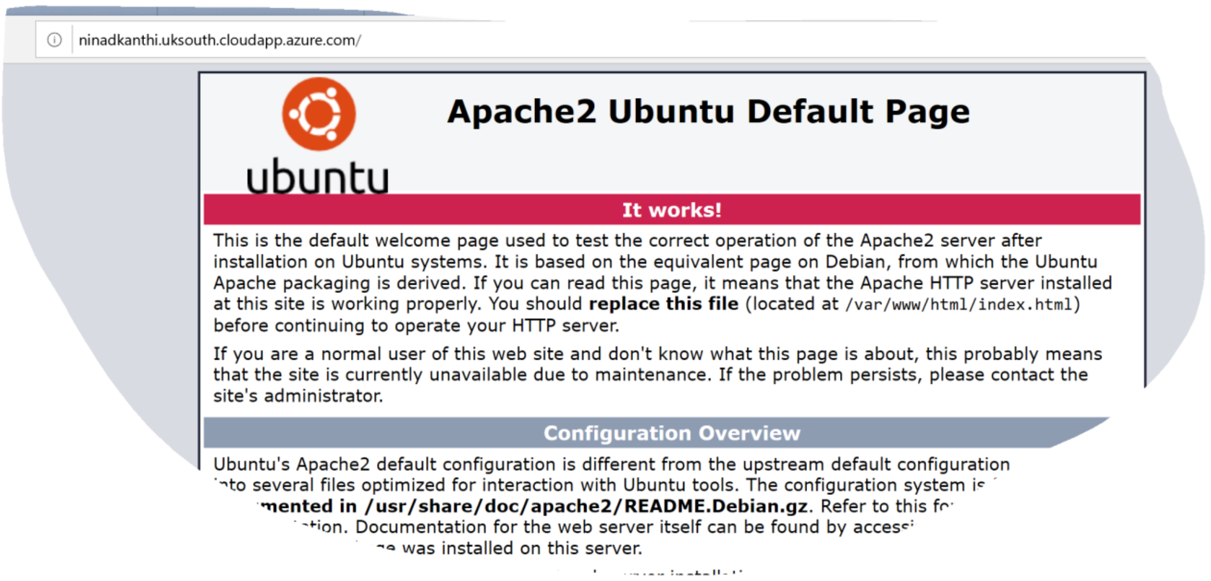
```bash
#!/bin/bash
# This script should be copied to webuser, chmod + x
# and then executed to check sudo permissions grant
sudo apt-get update
sudo apt-get upgrade
```

[VALIDATION TEST]: Successful execution of the script not only proves that the '**webuser**' has the requisite permissions to install downloads but also updates the system with the latest releases.

[AUTOMATION STEP]: Execute the script 'installApache.sh'. This script will install Apache server.

```bash
#!/bin/bash
sudo apt-get install apache2
sudo systemctl start apache2
sudo systemctl enable apache2
# check status
#sudo systemctl status apache2
```

[VALIDATION TEST]: Open browser and navigate to the public IP address of the Linux server. i.e. http://<IP_ADDRESS>. you should see the default Apache 2 Ubuntu default page like so


ninadkanthi.uksouth.cloudapp.azure.com/

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration into several files optimized for interaction with Ubuntu tools. The configuration system is mented in **/usr/share/doc/apache2/README.Debian.gz**. Refer to this fo tion. Documentation for the web server itself can be found by access e was installed on this server.

Static IP Address, DNS Name Label

Up until now we've provisioned connectivity for our custom domain name, i.e. *ninadkanthi.co.uk* from the external internet. Now we would like to establish the connectivity for the subdomain **blogs**.*ninadk.co.uk* with our Linux server. Two steps need to be done on the Azure side.

a) Provision static IP for the Linux server (along with DNS name label).

   a. **[AUTOMATION STEP]**: Execute the script 'update-public-IP-DNS-labelname.ps1' from the Windows PowerShell console. Script check if the Linux instance is running and if it is it stops it. Next it configures the Linux instance to have a static IP address as well as specifying a domain name label.

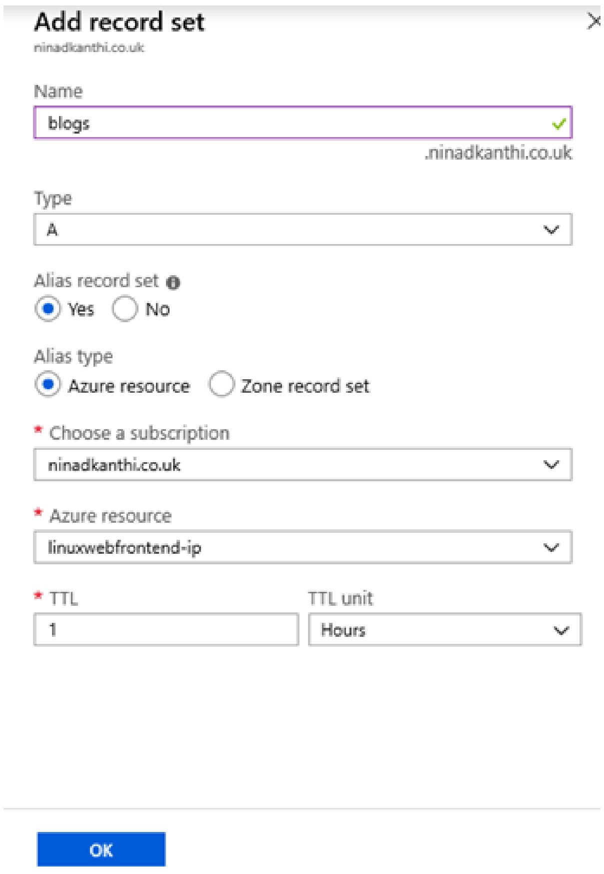
```
. "$PSScriptRoot\login.ps1"


$RunningState=((Get-AzureRmVM -ResourceGroupName
$RESOURCEGROUP_NAME -Name $virtualMachineName -
Status).Statuses[1]).Code
if ($RunningState -eq "Powerstate/Running")
{
    Write-Host "Stopping VM $virtualMachineName";
    $StoppedStatus= Stop-AzureRmVM -ResourceGroupName
$RESOURCEGROUP_NAME -Name $virtualMachineName
}

Write-Host "Updating new static IP address for
$virtualMachineName";
$publicIp = New-AzureRmPublicIpAddress  -Name $publicIpName -
ResourceGroupName $RESOURCEGROUP_NAME -AllocationMethod Static -
DomainNameLabel $dnsPrefix -Location $LOCATION

Write-Host "Starting $virtualMachineName ... ";
Start-AzureRmVM -ResourceGroupName $RESOURCEGROUP_NAME -Name
$virtualMachineName

Write-Host "... done";
```

   b. **[VALIDATION TEST]:** Validate that a successful result is returned. Open browser and navigate to the domain name label (**http://ninadkanthi.uksouth.cloudapp.azure.com**) specified in the previous script. It should display 'Your App service is running' message from Azure.

Install and Configure Azure DNS

This step will resolve the subdomain name to map to the IP address of the Linux server. It will provide the last mile connectivity.

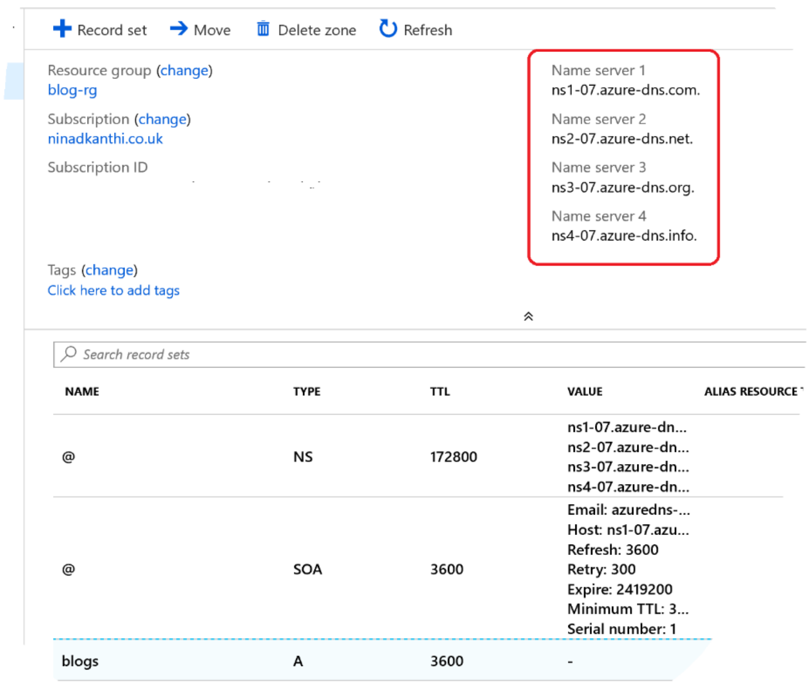
a) [AUTOMATION STEP]: Execute the script '[create-DNS-zone.ps1]' from the Windows PowerShell console. Validate that successful result was returned. Script creates a DNS zone under the Azure resource group.
b) [MANUAL STEP]: Login into the Azure portal and ensure that the DNS zone has been created. Select the DNS zone and manually add a DNS resource record (**+ Record Set**) as shown below.
    a. NOTE: in the following step, '**blogs**' is the subdomain specified, and the Azure resource is the public IP address of the Linux instance.



Add record set
ninadkanthi.co.uk

Name: blogs .ninadkanthi.co.uk
Type: A
Alias record set: [✓ Yes] No
Alias type: [✓ Azure resource] Zone record set
* Choose a subscription: ninadkanthi.co.uk
* Azure resource: linuxwebfrontend-ip
* TTL: 1
TTL unit: Hours

c) Whilst on the Azure portal, open the DNS Zone that was created. Make a note of the name servers - like so



Record set   Move   Delete zone   Refresh

Resource group (change)
blog-rg

Subscription (change)
ninadkanthi.co.uk

Subscription ID

Tags (change)
Click here to add tags

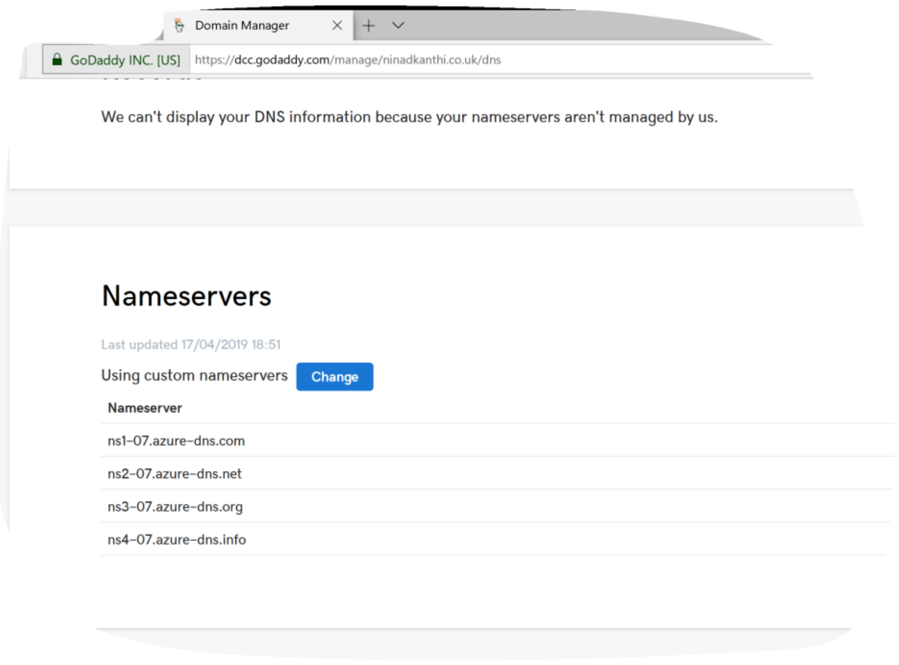
Name server 1	ns1-07.azure-dns.com.
Name server 2	ns2-07.azure-dns.net.
Name server 3	ns3-07.azure-dns.org.
Name server 4	ns4-07.azure-dns.info.

Search record sets

NAME	TYPE	TTL	VALUE	ALIAS RESOURCE
@	NS	172800	ns1-07.azure-dn... ns2-07.azure-dn... ns3-07.azure-dn... ns4-07.azure-dn...	
@	SOA	3600	Email: azuredns-... Host: ns1-07.azu... Refresh: 3600 Retry: 300 Expire: 2419200 Minimum TTL: 3... Serial number: 1	
blogs	A	3600	-	

d) Login to your domain registrar web site and change the default NS records specified there with the four records captured in the previous steps.

   a. In the GoDaddy.com site, the end-result should be as following:



Domain Manager

GoDaddy INC. [US] https://dcc.godaddy.com/manage/ninadkanthi.co.uk/dns

We can't display your DNS information because your nameservers aren't managed by us.

Nameservers

Last updated 17/04/2019 18:51

Using custom nameservers Change

Nameserver
ns1-07.azure-dns.com
ns2-07.azure-dns.net
ns3-07.azure-dns.org
ns4-07.azure-dns.info

e) [VALIDATION TESTS]: After ~ 1 hour, the changes should be reflected globally. Open the browser and navigate to your fully qualified domain name (FQDN) – **blogs.ninadkanthi.co.uk** It should display the default Apache Ubuntu page!

Install and configure MySQL Database

Login to the Linux server (as **webuser**) and execute following commands at the terminal prompt

```
~$ sudo apt-get install mysql-server
~$ mysql_secure_installation
~$ sudo systemctl start mysql
~$ sudo systemctl enable mysql
```

Script Note:

First command installs the mysql-server. During the installation of mysql, you'll be asked to enter password for 'root' user. Enter a strong password.

The second command – mysql_secure_installation – ensure that the database is configured with best practices for security.

Third and fourth commands are to start the database server and to enable it to automatically start upon boot.

Install and configure PHP

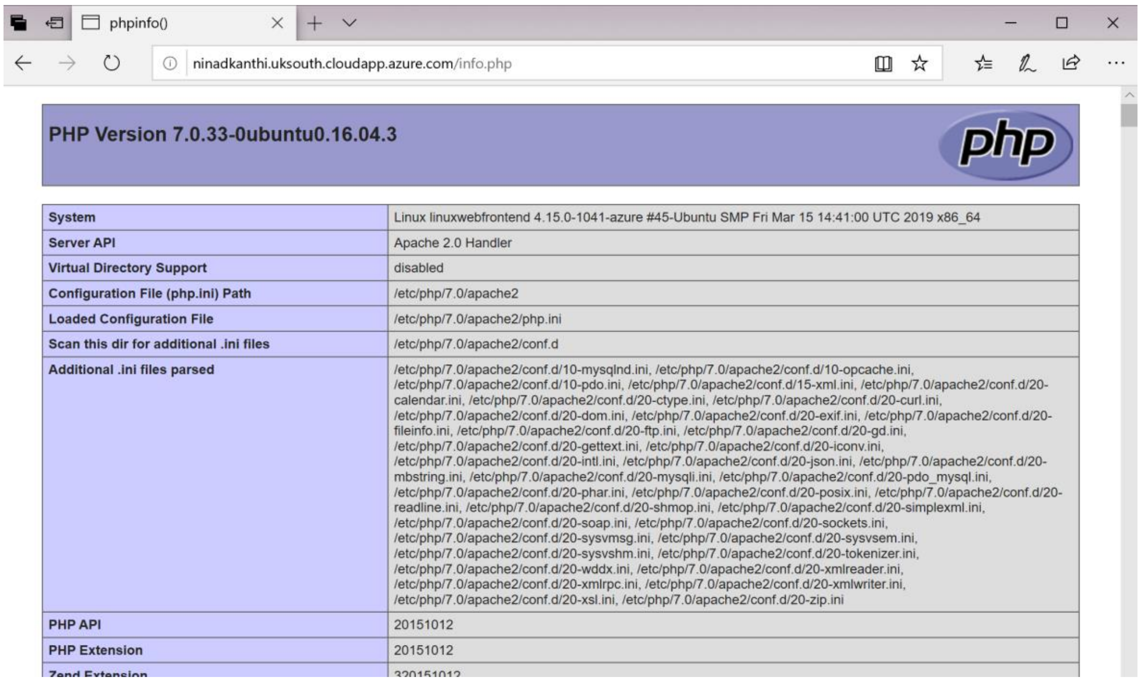
Login to the Linux server (as **webuser**) and execute following commands at the terminal prompt

```

~$ sudo apt-get install php7.0 libapache2-mod-php7.0 php7.0-mysql php7.0-curl php7.0-mbstring
php7.0-gd php7.0-xml php7.0-xmlrpc php7.0-intl php7.0-soap php7.0-zip

~$ sudo vi /var/www/html/info.php
~$ sudo cat /var/www/html/info.php
<?php
phpinfo();
?>
~$ sudo systemctl restart apache2
```

Command Note:

- First command installs PHP and its dependency modules.
- To test successful deployment of PHP, we'll now copy a file called info.pho into the Apache web server root directory (/var/www/html/). The content of the file is
  <?php> phpinfo(); ?>
- Second command show how to use the 'vi' editor and location where the file is created.
- Third command shows the content output of file.
- Fourth command re-starts the apache2 server

[Validation Tests]: Open the browser and navigate to your domain name label/info.php (**http://ninadkanthi.uksouth.cloudapp.azure.com/info.php**). It should now display the PHP configuration page instead of the default Apache Ubuntu page! Example is shown below.



PHP Version 7.0.33-0ubuntu0.16.04.3

System	Linux linuxwebfrontend 4.15.0-1041-azure #45-Ubuntu SMP Fri Mar 15 14:41:00 UTC 2019 x86_64
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.0/apache2
Loaded Configuration File	/etc/php/7.0/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.0/apache2/conf.d
Additional .ini files parsed	/etc/php/7.0/apache2/conf.d/10-mysqlnd.ini, /etc/php/7.0/apache2/conf.d/10-opcache.ini, /etc/php/7.0/apache2/conf.d/10-pdo.ini, /etc/php/7.0/apache2/conf.d/15-xml.ini, /etc/php/7.0/apache2/conf.d/20-calendar.ini, /etc/php/7.0/apache2/conf.d/20-ctype.ini, /etc/php/7.0/apache2/conf.d/20-curl.ini, /etc/php/7.0/apache2/conf.d/20-dom.ini, /etc/php/7.0/apache2/conf.d/20-exif.ini, /etc/php/7.0/apache2/conf.d/20-fileinfo.ini, /etc/php/7.0/apache2/conf.d/20-ftp.ini, /etc/php/7.0/apache2/conf.d/20-gd.ini, /etc/php/7.0/apache2/conf.d/20-gettext.ini, /etc/php/7.0/apache2/conf.d/20-iconv.ini, /etc/php/7.0/apache2/conf.d/20-intl.ini, /etc/php/7.0/apache2/conf.d/20-json.ini, /etc/php/7.0/apache2/conf.d/20-mbstring.ini, /etc/php/7.0/apache2/conf.d/20-mysqli.ini, /etc/php/7.0/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.0/apache2/conf.d/20-phar.ini, /etc/php/7.0/apache2/conf.d/20-posix.ini, /etc/php/7.0/apache2/conf.d/20-readline.ini, /etc/php/7.0/apache2/conf.d/20-shmop.ini, /etc/php/7.0/apache2/conf.d/20-simplexml.ini, /etc/php/7.0/apache2/conf.d/20-soap.ini, /etc/php/7.0/apache2/conf.d/20-sockets.ini, /etc/php/7.0/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.0/apache2/conf.d/20-sysvsem.ini, /etc/php/7.0/apache2/conf.d/20-sysvshm.ini, /etc/php/7.0/apache2/conf.d/20-tokenizer.ini, /etc/php/7.0/apache2/conf.d/20-wddx.ini, /etc/php/7.0/apache2/conf.d/20-xmlreader.ini, /etc/php/7.0/apache2/conf.d/20-xmlrpc.ini, /etc/php/7.0/apache2/conf.d/20-xmlwriter.ini, /etc/php/7.0/apache2/conf.d/20-xsl.ini, /etc/php/7.0/apache2/conf.d/20-zip.ini
PHP API	20151012
PHP Extension	20151012
Zend Extension	320151012

Finally remove the file as otherwise it will provide too much information to the user about your systems to outside user. Execute following command

```
~$ sudo rm /var/www/html/info.php
```

Install and configure WordPress

Login to the Linux server (as **webuser**) and execute following commands at the terminal prompt

```
1) ~$ cd /var/www/html
2) :/var/www/html $ sudo wget -c http://wordpress.org/latest.tar.gz
3) :/var/www/html $ sudo tar -xzvf latest.tar.gz
4) :/var/www/html $ ls ./wordpress/
5) :/var/www/html $ sudo chown -R www-data:www-data /var/www/html/wordpress
6) :/var/www/html$ mysql -u root -p
…
6a) mysql> CREATE DATABASE wordpress_db;
6b) mysql> GRANT ALL PRIVILEGES ON wordpress_db.* TO 'webuser'@'localhost'
IDENTIFIED BY ‘STRONG_PASSWORD’;
6c) mysql> FLUSH PRIVILEGES;
6d) mysql> exit
7) :/var/www/html$ cd wordpress/
8) :/var/www/html/wordpress$ sudo cp ./wp-config-sample.php wp-config.php
9) :/var/www/html/wordpress$ sudo vi wp-config.php
…
/** The name of the database for WordPress */
9a) define( 'DB_NAME', wordpress_db);
/** MySQL database username */
9b) define( 'DB_USER', 'webuser' );
/** MySQL database password */
9c) define( 'DB_PASSWORD', ' STRONG_PASSWORD’' );
…
10) :/var/www/html/wordpress$  sudo systemctl restart mysql
11) :/var/www/html/wordpress$  sudo systemctl restart apache2
```

Command Note:

1) Move to the default web server document root directory (/var/www/html). This is where we'd copy the download and install WordPress
2) Download latest version of WordPress in zipped format
3) Decompress and extract the content of the downloaded tar file
4) Manually confirm and check that output is non-empty.
5) Need to set correct permissions of this directory so the Apache server can access these files
6) Create and configure mysql database that will be used by WordPress. Make note of the name of the database ‘wordpress_db’, user name ‘webuser’ and password ‘STRONG_PASSWORD’. Substitute ‘STRONG_PASSWORD’ with a strong password.
7) Make WordPress web base folder as current directory
8) Make a copy of the sample web configuration file added during the installation and rename it to be the default configuration file of WordPress
9) Open the web configuration file and substitute the database name, username and password noted previously. Save and close the file
10) Restart mysql
11) Restart Apache2

[VALIDATION TESTS]: To test the installation, launch browser app, open the default url with ‘wordpress’ appended at the end. For the first time launch, it'll ask you to create a wed admin username and password. Once logged in, it'll redirect to its admin web page -

http://blogs.ninadkanthi.co.uk/wordpress/wp-login.php. You can now make changes to the WordPress website using the administrator portal.

Congratulations, you've successfully installed WordPress and its also connected to the whole internet!

Replacing default apache2 and WordPress web pages.

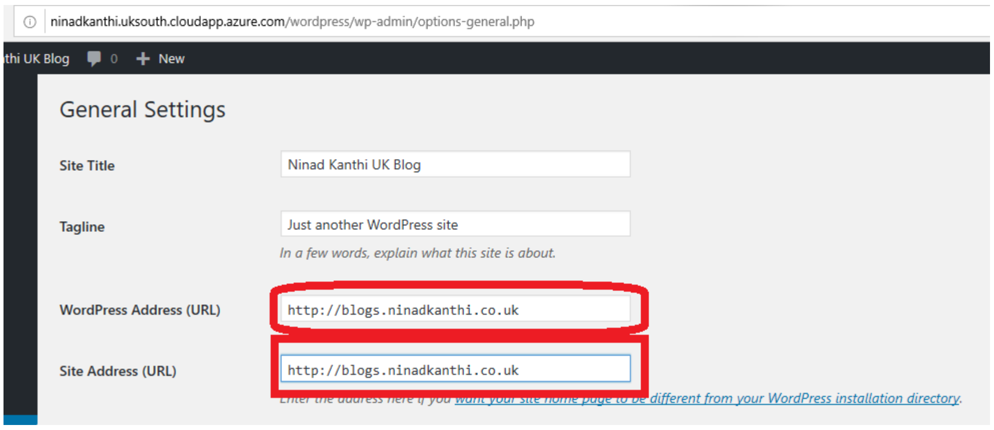
At this point, if we navigate to 'http://blogs.ninadkanthi.co.uk/wordpress' we are taken to the WordPress web site default page. But if we navigate to any of the following URLs - http://<IPaddress-of-Linux-machine> or http://ninadkanthi.uksouth.cloudapp.azure.com/ or http://blogs.ninadkanthi.co.uk/ - they all show the default Apache 2 welcome page.

In this section, we'll configure WordPress and Apache2 such that both display the WordPress blogs home page when navigating to above URLs.

WordPress configurations changes

Login to the WordPress admin portal - http://blogs.ninadkanthi.co.uk/wp-login.php --> http://ninadkanthi.uksouth.cloudapp.azure.com/wordpress/wp-admin/options-general.php

Change the two entries from original to new fully qualified web site address: The new values should look as following.



ninadkanthi.uksouth.cloudapp.azure.com/wordpress/wp-admin/options-general.php

General Settings

Site Title: Ninad Kanthi UK Blog

Tagline: Just another WordPress site

In a few words, explain what this site is about.

WordPress Address (URL): http://blogs.ninadkanthi.co.uk

Site Address (URL): http://blogs.ninadkanthi.co.uk

Enter the address here if you want your site home page to be different from your WordPress installation directory.

Click Save Changes.

Apache2 configuration changes

To configure Apache2, execute following commands at the terminal prompt.

*Pre-step: Copy script 'replace_str.sh' & 'py_file_sub_str.py' onto the Linux machine and move the scripts to the folder **/etc/apache2/** & **/etc/apache2/sites-available/** respectively.*

```
1) ~ $ cd /etc/apache2/
2) :/etc/apache2$ sudo chmod +x ./replace_str.sh
3) :/etc/apache2$ ./replace_str.sh
4) :/etc/apache2$ sudo service apache2 restart
5) :/etc/apache2$ cd sites-available/
6) :/etc/apache2/sites-available$ sudo cp 000-default.conf wordpress.conf
7) :/etc/apache2/sites-available$ sudo python py_file_sub_str.py
8) :/etc/apache2/sites-available$ sudo a2dissite 000-default.conf
9) :/etc/apache2/sites-available$ sudo a2ensite wordpress.conf
10):/etc/apache2/sites-available$ sudo service apache2 restart
11):/etc/apache2/sites-available$ sudo service apache2 reload
```

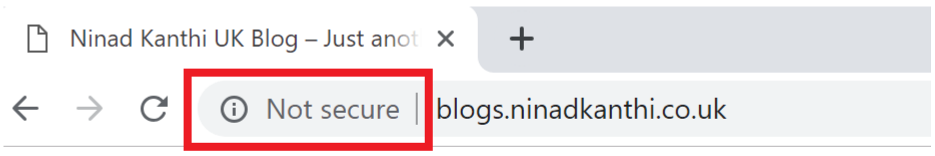
Commmand Notes:

1) Set current working directory to apache2 installation folder
2) Mark the previously copied script ‘replace_str.sh’ as executable
3) Execute the script. It changes the default web site location from ‘/var/html/’ to ‘/var/www/html/wordpress’
4) Restart the Apache service. Note that no error is displayed
5) Change the current working directory to **/etc/apache2/sites-available**
6) Make a copy of default web configuration file
7) Execute script ‘**py_file_sub_str.py**’ using python. Substitute relevant code from original to indicate **wordpress** web site as default. See ‘**py_file_sub_str.py**’ script for the couple of substitutions that it does.
8) Remove default configuration as the enabled web site configuration
9) Make the newly created **wordpress.conf** as the enabled web site configuration
10) Restart Apache2
11) Reload Apache2

[VALIDATION TESTS]: Opening the URL - http://blogs.ninadkanthi.co.uk/ or http://ninadkanthi.uksouth.cloudapp.azure.com/ should now display the first page of the blog web site. Directly accessing the IP address should now give a 404 error(Web site not found error!)

Configuring SSH

Any end user opening and viewing our blogs web site is shown ‘not secure’ warning. Like so:



This section we’d install certificates to ensure that this **Not secure** warnings goes away.

Installing Certificates

To install SSH certificates, execute following commands at the terminal prompt

Pre-step: Copy the script 'installcertificates.sh' onto the Linux machine. Update the script with the custom domain name in the line

```
1) ~ $ sudo chmod +x ./installcertificates.sh
2) ~ $ ./installcertificates.sh
```

Command Notes:

1) Mark the script as executable
2) Execute the script. Script used www.letsencrypt to install certificates. Ensure that you've specified the correct domain name to the script and that command can access and install files to the machine.

[Validation Tests]: After the successful execution of the script, README file is displayed from the installation folder.

Apache2 SSH configuration

Pre-step: Copy the script 'py_file_replace_str.py' and 'configure-ssl.sh' files onto the Linux machine. Copy both the files to folder **/etc/apache2/sites-available.** *Edit configure-ssl.sh script to reflect your custom domain name and SSH keys.*

Execute the following commands

```
1):~$ cd /etc/apache2/sites-available
2):/etc/apache2/sites-available$ sudo chmod +x ./configure-ssl.sh
3) :/etc/apache2/sites-available$ sudo cp default-ssl.conf wordpress-ssl.conf
4) :/etc/apache2/sites-available$ ./configure-ssl.sh
5) :/etc/apache2/sites-available$  sudo apachectl configtest
6) :/etc/apache2/sites-available$ sudo a2enmod ssl
7) :/etc/apache2/sites-available$ sudo a2ensite wordpress-ssl.conf
8) :/etc/apache2/sites-available$ sudo service apache2 restart
```

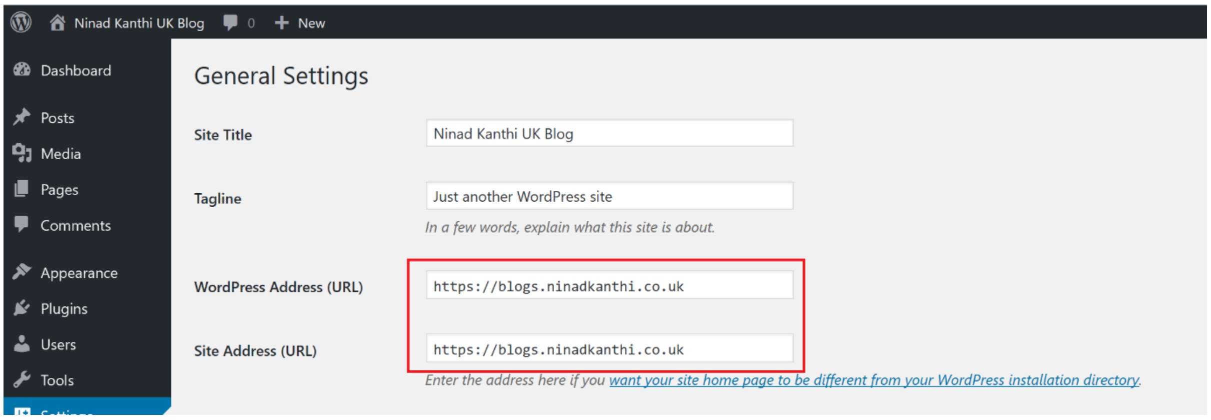
Command Notes:

1) Change the current working directory to **/etc/apache2/sites-available**
2) Mark the script as executable
3) Copy the default ssl configuration into a new file called **wordpress-ssl.conf**
4) Execute the script. It replaces the SSH key information and custom domain names.
5) Test that the configuration added is correct
6) Enable SSL for the apache web site
7) Enable the SSL for the web site and specify the configuration file as the one that was newly created – wordpress-ssl.conf.
8) Restart Apache2 web site for the changes and SSL to be available.

[Validation Tests]: After this step, you should be able to your web site with not only http prefix but also https prefix

WordPress SSH configuration

Login to the web administrator portal (http://blogs.ninadkanthi.co.uk/wp-admin/options-general.php). Change the default configuration to use HTTPS



General Settings

Site Title: Ninad Kanthi UK Blog

Tagline: Just another WordPress site

In a few words, explain what this site is about.

WordPress Address (URL): https://blogs.ninadkanthi.co.uk

Site Address (URL): https://blogs.ninadkanthi.co.uk

Enter the address here if you want your site home page to be different from your WordPress installation directory.

Click Save changes when done

HTTPS Redirect.

When the user logs in using http, we need to re-direct them to use the HTTPS instead of HTTP.

Add the following highlighted line to file **/etc/apache2/sites-available/wordpress.conf**

```
…

        #Include conf-available/serve-cgi-bin.conf
        Redirect / https://blogs.ninadkanthi.co.uk
</VirtualHost>
```

After this restart and reload apache2 like you've done before. Everything now will be running in https.

NOTE: When you renew the certificates, you'll need to disable the redirect added above.

At this point, your blogs web site is up and running and accepting HTTPS traffic!