LAMP Stack with VirtualHosts On Centos 6.x

This article illustrates how to install the Apache Mysql PHP Stack on Centos 6.x.

Additionally, with this configuration, you can serve Multiple Domains using the Virtual Hosts Apache directive.

Install Apache

Invoke yum for installation of Apache

```
yum install -y httpd mod_ssl httpd-devel
```

@!:{httpd-devel libraries were included in order to have module compile capabilities, as well as being able to install modules from source

Enable autostart of the Apache service

```
chkconfig httpd on
```

Start the Apache service

```
service service httpd resart
```

Install PHP

Install PHP, et al

```
yum install -y php php-mysql php-common php-mbstring php-mcrypt php-devel php-xml php-pecl-memcache php-pspell php-snmp php-xmlrpc php-gd
```

Restart the Apache service

```
service httpd restart
```

Check DNS

Ensure there exists a DNS entry for the domain you want to use.

If this is a lab setup, or completely local, you can simply create a hosts entry for the domain, e.g.

```
vi /etc/hosts
```

[divider]

Virtual Hosts

The **NameVirtualHost** directive allows us to host multiple websites on a single web server.

Example:

You want to host **mydomain1.com** on your web server
You also want to host **mydomain2.com** on your web server

In order to accomplish this, you'll need to:
– enable the NameVirtualHost directive
– create appropriate configuration files for the domains in question, e.g.:

```
/etc/httpd/conf.d/mydomain1.com.conf
/etc/httpd/conf.d/mydomain2.com.conf
```

For now, let's configure just one domain, *mydomain1.com*:

[divider]

Create Vhosts Config Directories

```
Create a vhost config folder
```

```
mkdir -p /etc/httpd/vhost.d
```

Configure NameVirtualHost Directive

Add an include directive to the apache config file:

```
vim /etc/httpd/conf/httpd.conf
    Include vhost.d/*.conf
```

@!:{The above makes it so that any files ending in .conf under the folder vhost.d are included as part of the httpd.conf configuration
Notice that ***vhost.d*** is a relative path. The full path would be evaluated as ServerRoot/vhost.d, where ServerRoot is /etc/httpd (see the httpd.conf file for more information)

Comment out any Listen directives and add an include directive to a separate ports settings config file:

```
#Listen 12.34.56.78:80
#Listen 80
Include ports.conf
```

@!:{The above makes it so that the ports.conf file is included as part of the httpd.conf configuration
What this accomplishes is a separation of port specification from the main config file

Create a ports config file

```
vi /etc/httpd/ports.conf
```

With contents:

```
Listen $Port
NameVirtualHost $IPPUBLIC:$Port
NameVirtualHost $IPPRIVATE:$Port
```

```
NameVirtualHost *:$Port
```

Where **$Port** is the numeric value of the port number through which you want Apache to listen for traffic

```
#e.g.
NameVirtualHost 192.168.250.188:80
NameVirtualHost 127.0.0.1:80
NameVirtualHost *:80
```

Restart Apache

```
service httpd restart
```

Create The Config File for the Virtual Host/Domain

Create a config file for your domain

```
vim /etc/httpd/vhost.d/mydomain1.conf
```

```
   <VirtualHost *:80>

    ServerName mydomain1.com
    ServerAlias www.mydomain1.com
    DocumentRoot /var/www/vhosts/mydomain1.com
    <Directory /var/www/vhosts/mydomain1.com>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All
    </Directory>

       CustomLog  /var/log/httpd/mydomain1.com-access.log
combined
    ErrorLog /var/log/httpd/mydomain1.com-error.log
```

```
    # Possible values include: debug, info, notice, warn,
error, crit,
    # alert, emerg.
    LogLevel warn

    </VirtualHost>
```

Make sure your document root exists!

```
mkdir /var/www/vhosts/mydomain1.com
#—OR Try this One-liner—#
ls /var/www/vhosts/mydomain1.com 2> /dev/null || echo does
not exist;echo creating folder;mkdir -p
/var/www/vhosts/mydomain1.com && echo created folder!
```

[divider]

Modify Firewall

You'll need to poke a hole in the firewall to allow communication to the Apache listening port (by default port 80):
Edit iptables config

```
vi /etc/sysconfig/iptables
A INPUT -m state —state NEW -m tcp -p tcp —dport 80 -j ACCEPT
```

Restart iptables

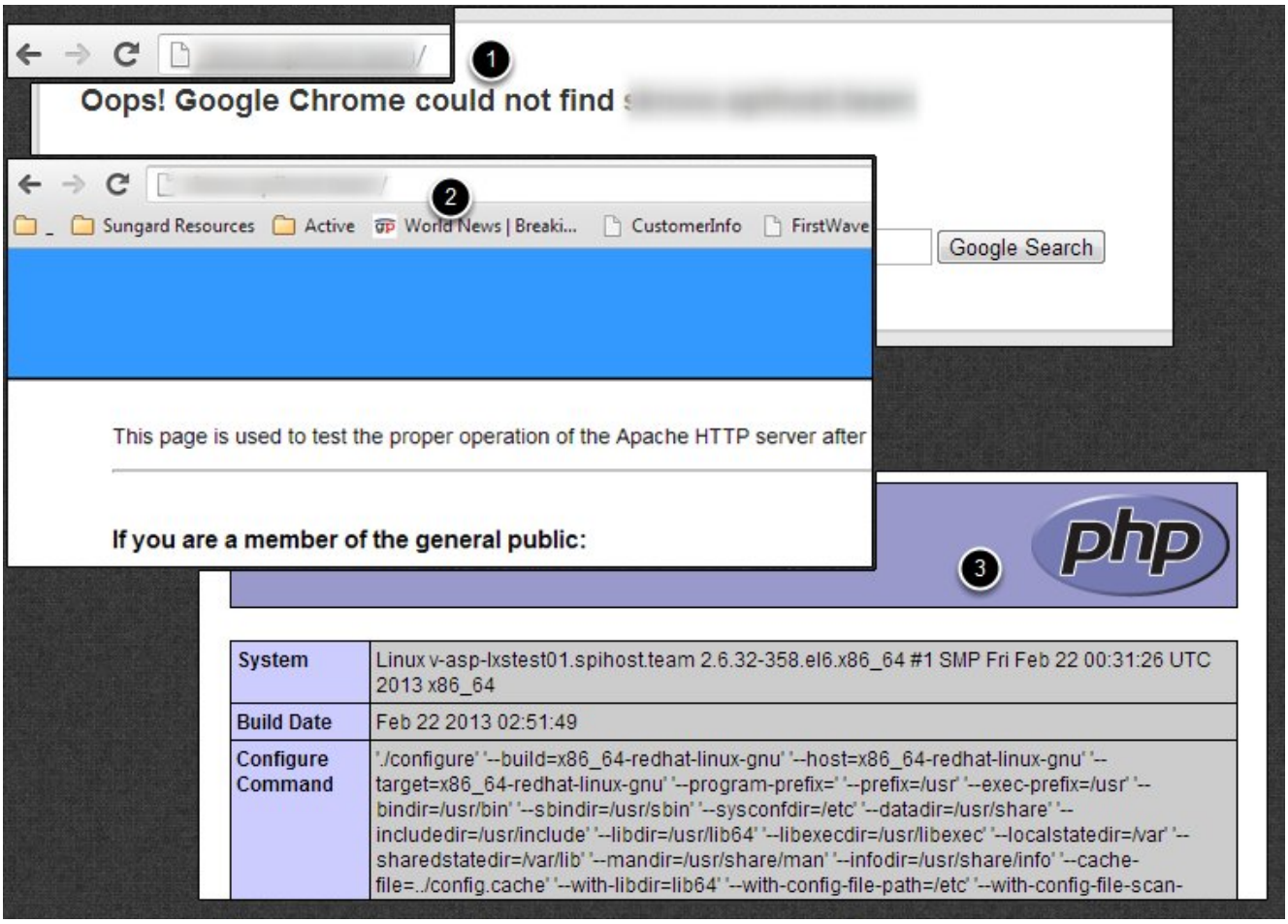
```
service iptables restart
```

[divider]

Troubleshooting

[divider]

Error – Could not find …



1. Problem: When navigating to your domain via web browser, you receive an error similar to ‘could not find’

Q:{Is DNS setup correctly?
Check:

```
nslookup mydomain1.com
```

if error then ensure DNS record exists on your DNS server

if Windows, try the *ipconfig /flushdns* command

Q:{Is Firewall to blame?
Check:

```
telnet $yourdomain $port
```

e.g.

```
telnet mydomain1.com 80
```

if error then ensure Firewall port is open:

```
vi /etc/sysconfig/iptables
e.g. -A INPUT -m state –state NEW -m tcp -p tcp –dport 80 -j
ACCEPT
```

Restart firewall:

```
service iptables restart
```

2. Test website access again
Hopefully Success!

3. Test PHP functionality:

```
vi /var/www/vhosts/domain.com/index.php
```

```
<?php
phpinfo();
?>
:wq
```

Test website access again

```
http://mydomain1.com/index.php
```

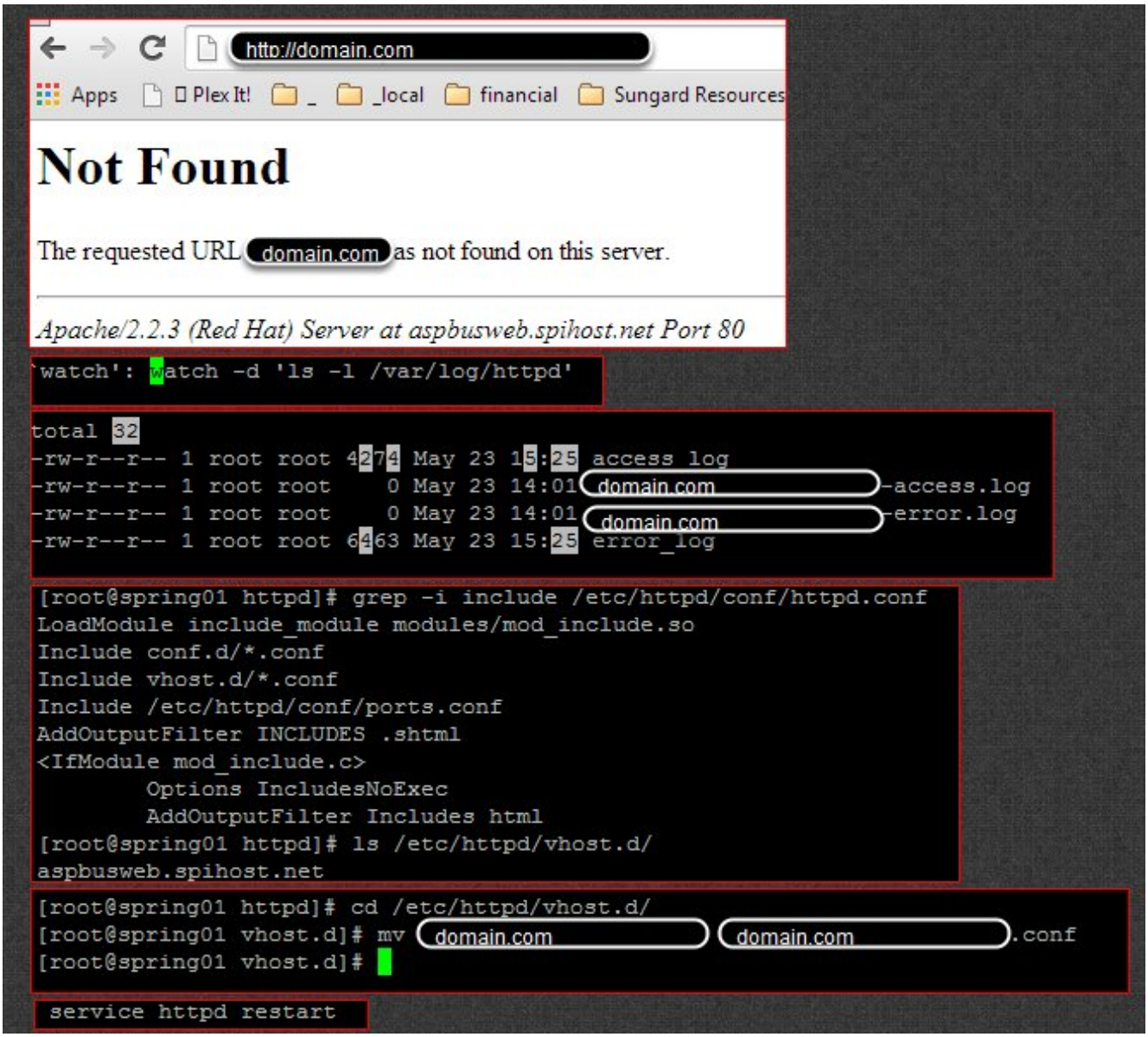
If you've made numerous changes, try restarting the Apache service again

```
service httpd restart
```

If all else fails, and if you have the option to do so, reboot the server

```
reboot
```

Error – requested URl was not found on this server



In this case, I created the config file for the domain under vhosts.d, but had forgotten to give it a .conf file extension. doh!

Note how I used the watch command to ‘watch’ for changes to log files under /var/log/httpd.

This functions much like inotifywait for troubleshooting using log files.

Automating IIS

This Article presents some examples of automating IIS using PowerShell and the appcmd.exe tool

References I used

Title	URL IIS AppCmd Quick Reference	http://blogs.msdn.com/b/mikezh/archive/2012/04/23/iis-appcmd-quick-reference.aspx ""	https://github.com/jzabroski/OrchardBootstrap/blob/master/OrchardBootstrap.psi PowerShell Snap-in	http://www.iis.net/learn/manage/powershell/powershell-snap-in-creating-web-sites-web-applications-virtual-directories-and-application-pools

PowerShell Commands

This applies to my environment: Before you can utilize PowerShell commands for IIS automation, you'll need to ensure two things:

- PowerShell Script Execution is allowed for your session

[code language="powershell"]Set-Executionpolicy Bypass -Scope Process[/code]

- IIS Web Administration PowerShell module is properly loaded

[code language="powershell"]Import-Module WebAdministration[/code]

In the following end-to-end scenario we will execute the following steps:

1. Create new Application Pool
2. Create a new site
3. Create a new application
4. Assign the newly created Application to the Newly Created AppPool
5. Create two virtual directories
6. Assign permissions to these virtual directories

No.	Action	Command								
	Create an IIS AppPool named MyAppPool	New-Item AppPools\MyAppPool	Create an IIS Site named MySite with a physical path of C:\MySite	New-Item IIS:\Sites\MySite -bindings @{protocol="http";bindingInformation=":80:MySite"} -physicalPath C:\MySite	Create an IIS Application named MyApplication with a physical path of C:\MySite\MyApplication	New-Item 'IIS:\Sites\MySite\MyApplication' -physicalPath C:\MySite\MyApplication -type Application	Modify Application – Set AppPool for MyApplication to MyAppPool	Set-ItemProperty "IIS:\Sites\MySite\MyApplication" -name ApplicationPool -value MyAppPool	Create a Virtual Directory MyVirtualDirectory under MySite\MyApplication pointing to C:\MySite\MyApplication\MyVirtualDirectory	New-Item 'IIS:\Sites\MySite\MyApplication\MyVirtualDirectory' -type VirtualDirectory -physicalPath "C:\MySite\MyApplication\MyVirtualDirectory"	Assign permissions for MyApplication\MyVirtualDirectory	$file = $(get-item "IIS:\Sites\MySite\MyApplication\App_Data\") $dacl = $file.GetAccessControl() $newRule = New-Object Security.AccessControl.FileSystemAccessRule "BUILTIN\IIS_IUSRS", Write, Allow $modified = $false $dacl.ModifyAccessRule("Add", $newRule, [ref]$modified) $file.SetAccessControl($dacl) $file.GetAccessControl().GetAccessRules($true, $true, [System.Security.Principal.NTAccount])

Additional Commands

Action	Command									
Copy IIS AppPool MyAppPool to MyNewAppPool	forcing overwrite	Copy IIS:\AppPools\MyAppPool IIS:\AppPools\MyNewAppPool –force	Remove the site	Remove-Item IIS:\Sites\MySite	Copy IIS Application MyApplication to MyNewApplication	forcing overwrite	Copy "IIS:\Sites\MySite\MyApplication" "IIS:\Sites\MySite\MyNewApplication" -force	Start IIS MySite	Start-WebItem 'IIS:\Sites\MySite'	Stop IIS MySite	Stop-WebItem 'IIS:\Sites\MySite'	Start IIS Application Pool	Start-WebAppPool -Name "MyAppPool"	Stop IIS Application Pool	Stop-WebAppPool -Name "MyAppPool"	Get Application Pool Status	Get-WebAppPoolState $appPoolName

Note: For the sake of example: Application Pool Name is **MyAppPool** and Site Name is **MySite**

IIS AppCmd Quick Reference

Action	Command Add Site											
/providers:[value='Negotiate'] /commit:apphost (set to Negotiate) Add Custom Header – for example, noscliff header or IE 7 compatible header	appcmd set config MySite -section:system.webServer/httpProtocol /-	appcmd add site /name:MySite /bindings:http://*:80 /physicalpath:"d:\MySite\wwwroot" Add App Pool	appcmd add apppool /name:MyAppPool /managedRuntimeVersion:v4.0 (e.g. targeting .NET 4.0) Set App Pool Credential	appcmd set config /section:applicationPools /[name='MyAppPool'].processModel.identityType:SpecificUser /[name='MyAppPool'].processModel.userName:MyDomain\MyAccount /[name='MyAppPool'].processModel.password:MyAccountPassword Add App	appcmd add app /site.name:"MySite" /path:/MyApp /physicalpath:"d:\MySite\MyApp\" Assign/Change App Pool to an App	appcmd set app "MySite/MyApp" /applicationpool:MyAppPool List (App, Site, AppPool, etc.)	appcmd list app appcmd list site appcmd list apppool /enabled:Disable Anonymous Authentication (True to Enable, False to Disable)	appcmd set config "MySite/MyApp" -section:system.webServer/security/authentication/anonymousAuthentication /enabled:"true" /commit:apphost Enable Windows Authentication (True to Enable, False to Disable)	appcmd.exe set config "MySite/MyApp" -section:system.webServer/security/authentication/windowsAuthentication /enabled:"true" /commit:apphost Change Windows Authentication Providers (NTLM or Negotiate)	appcmd set config MySite/MyApp -section:system.webServer/security/authentication/windowsAuthentication /-providers /commit:apphost (clear provider list) appcmd set config MySite/MyApp -section:system.webServer/security/authentication/windowsAuthentication /+providers.[value='NTLM'] /commit:apphost (set to NTLM) appcmd set config MySite/MyApp -section:system.webServer/security/authentication/windowsAuthentication /-
customHeaders.[name='X-Content-Type-Options',value='nosniff'] /commit:apphost												
appcmd set config MySite -section:system.webServer/httpProtocol /-												
customHeaders.[name='X-UA-Compatible',value='IE=8&ctrl=3'] /commit:apphost Add Default Document – error if it exists already	appcmd set config "MySite/MyApp" /section:defaultDocument /-											
files.[value='default.aspx'] Delete App and Site – error if it doesn't exist	appcmd delete app "MySite/MyApp" appcmd delete site "MySite" Delete AppPool – error if it doesn't exist or it is used by app	appcmd delete apppool MyAppPool Backup and Restore IIS Settings	appcmd add backup MyBackup MyBackup HTTPS Binding if you are using HTTP over SSL	appcmd set site /site.name:"MyApp" /								
bindings.[protocol='https',bindingInformation='*:443:MySSLCertificate']												