VDS Server Setup Documentation

- Security
    - Firewall configuration
    - Create Administrative users
    - Remove default user
- Base Software
    - Remove pre-installed software
    - Install Apache
    - Configure Apache
    - Install & Configure vnStat (optional)
    - Install PHP
    - Configure PHP
    - Install Java JDK
    - Install Tomcat
    - Configure Tomcat
    - Install GIT
    - Configure GIT
    - Add GIT Users
    - Install Jenkins
    - Configure Jenkins
- Test Application (Apache & PHP)
- Test Application (Apache & Tomcat)

Firewall Configuration

Log in as root

su root

iptables

vi /etc/sysconfig/iptables

Create the firewall configuration file:

vi /etc/sysconfig/iptables

Replace contents with:

```
# Flush & Delete Chains
*filter

# Set Chain Policy
:FORWARD DROP [0:0]
:INPUT DROP [0:0]
:OUTPUT DROP [0:0]

# Loopback (127.0.0.1)
-A INPUT -i lo -j ACCEPT
-A OUTPUT -o lo -j ACCEPT

# Outgoing HTTP
-A OUTPUT -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT
-A INPUT -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT

# Outgoing HTTPS
-A OUTPUT -p tcp --dport 443 -m state --state NEW,ESTABLISHED -j ACCEPT
-A INPUT -p tcp --sport 443 -m state --state ESTABLISHED -j ACCEPT

# Outgoing HTTPS
-A OUTPUT -p tcp --dport 443 -m state --state NEW,ESTABLISHED -j ACCEPT
-A INPUT -p tcp --sport 443 -m state --state ESTABLISHED -j ACCEPT

# Outgoing DNS
-A OUTPUT -p udp --dport 53 -j ACCEPT
-A INPUT -p udp --sport 53 -j ACCEPT

# Outgoing sendmail
-A INPUT -p tcp --dport 25 -m state --state NEW,ESTABLISHED -j ACCEPT
-A OUTPUT -p tcp --sport 25 -m state --state ESTABLISHED -j ACCEPT

# Incoming SSH
-A INPUT -p tcp --dport 22 --sport 513:65535 -m state --state NEW,ESTABLISHED -j ACCEPT
-A OUTPUT -p tcp --sport 22 --dport 513:65535 -m state --state ESTABLISHED -j ACCEPT
#-A INPUT -p tcp --dport 443 -m limit --limit 25/minute --limit-burst 100 -j ACCEPT

# Incoming HTTP
-A INPUT -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT
-A OUTPUT -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT
#-A INPUT -p tcp --dport 80 -m limit --limit 25/minute --limit-burst 100 -j ACCEPT

# Incoming HTTPS
-A INPUT -p tcp --dport 443 -m state --state NEW,ESTABLISHED -j ACCEPT
-A OUTPUT -p tcp --sport 443 -m state --state ESTABLISHED -j ACCEPT
#-A INPUT -p tcp --dport 443 -m limit --limit 25/minute --limit-burst 100 -j ACCEPT
```

```
# Drop all other packets
-A INPUT -j DROP
-A OUTPUT -j DROP

COMMIT
```

Restart the firewall

/etc/init.d/iptables restart

Log out of root

exit

Create administrative users (confirmed)

Switch to root user

su root

Create our admin group

/usr/sbin/groupadd admin

Create our admin users

/usr/sbin/adduser -n -g admin ksimons

Setup our admin passwords

/usr/bin/passwd ksimons

Allow the admin group to sudo

/usr/sbin/visudo

Scroll to the bottom and add the following line to allow the admin group to have full sudo access

```
%admin  ALL=(ALL)       ALL
```

Modify SSH configuration

vi /etc/ssh/sshd_config

Ensure PermitRootLogin is set to 'no'

```
PermitRootLogin no
```

Remove the DenyUsers & DenyGroups entries (Since we will be using whitelist, not blacklist)

```
DenyUsers root
DenyGroups root
```

Add our white list entries

```
AllowGroups admin <DEFAULT_GROUP>
```

Tell SSH to reload its config file

/etc/rc.d/init.d/sshd reload

Exit our superuser (root) login

exit

Remove default user (confirmed)

Login to an admin account we created in the previous section. We will need to confirm we have ssh/sudo access before we can delete our default user

Ensure you have sudo access

sudo -u root /etc/rc.d/init.d/sshd status

If you have sudo access, open the SSH config file again

sudo vi /etc/ssh/sshd_config

Remove the <DEFAULT_GROUP> from the following line:

```
AllowGroups admin <DEFAULT_GROUP>
```

Tell SSH to reload its config file

sudo /etc/rc.d/init.d/sshd reload

Remove the default user (and its data) we were initially logged in as

sudo /usr/sbin/userdel -rf monkey

Remove the group the default user was a part of (this should have been done automatically above, however)

sudo /usr/sbin/groupdel monkey

Remove pre-installed software (confirmed)

Remove the mysql package

sudo yum remove mysql.i386

Yum does not remove the user/group, do it manually

sudo /usr/sbin/userdel mysql
sudo /usr/sbin/groupdel mysql

Remove the php package

sudo yum remove php.i386 php-*

Yum does not remove the folders/files

sudo rm -rf /etc/php.d
sudo rm -rf /usr/lib/php
sudo rm -rf /var/lib/php

Remove the httpd package

sudo yum remove httpd.i386

Yum does not remove the user name, do it manually

sudo /usr/sbin/userdel apache
sudo /usr/sbin/groupdel apache

Yum does not remove the folders either

sudo rm -rf /etc/httpd
sudo rm -rf /usr/lib/httpd
sudo rm -rf /var/log/httpd

Install Apache (confirmed)

Install apache from our yum repository

sudo yum install httpd.i386

Install mod_ssl from our yum repository

sudo yum install mod_ssl.i386

Download mod_jk from the apache website

wget http://archive.apache.org/dist/tomcat/tomcat-connectors/jk/binaries/linux/jk-1.2.31/i386/mod_jk-1.2.31-httpd-2.2.x.so

Move the module to the the correct folder

sudo mv mod_jk-1.2.31-httpd-2.2.x.so /etc/httpd/modules/

Delete welcome.conf

sudo rm /etc/httpd/conf.d/welcome.conf

Disable ssl on the default virtual host

sudo vi /etc/httpd/conf.d/ssl.conf

Remove the virtualhost at the end of the file

```
<VirtualHost _default_:443>
            ...
</VirtualHost>
```

Disable proxy_ajp

sudo mv /etc/httpd/conf.d/proxy_ajp.conf /etc/httpd/conf.d/proxy_ajp.conf.disabled

Make sure httpd is started after reboot

sudo /sbin/chkconfig --level 2345 httpd on

Restart apache

sudo /etc/init.d/httpd restart

Configure Apache (confirmed)

Modify the default apache configuration file

sudo vi /etc/httpd/conf/httpd.conf

For future changes

```
...
```

Create our mod_jk configuration file

sudo vi /etc/httpd/conf.d/mod_jk.conf

The file should contain the following:

```
LoadModule jk_module modules/mod_jk-1.2.31-httpd-2.2.x.so

JkWorkersFile conf.d/workers.properties
JkLogFile     logs/mod_jk.log
JkLogLevel            info
JkLogStampFormat "[%a %b %d %H:%M:%S %Y] "
JkOptions     +ForwardKeySize +ForwardURICompat -ForwardDirectories
JkRequestLogFormat     "%w %V %T"
```

Create our skeleton mod_jk worker properties file

sudo vi /etc/httpd/conf.d/workers.properties

The file should contain the following:

```
# Define the workers
worker.list=
```

Create our base virtual host configuration file

sudo vi /etc/httpd/conf.d/vhost.conf

The file should contain the following:

```
NameVirtualHost *:80

# Since it is our first virtual host, it will catch all requests not caught by subsequent virtual hosts
<VirtualHost *:80>
        # Inherit default values from httpd.conf
        ServerName localhost
</VirtualHost>
```

Restart apache

sudo /etc/init.d/httpd restart

Modify the group for /var/www

sudo chgrp -R admin /var/www/html

Allow the group to write

sudo chmod -R g+w /var/www/html

Create a folder to hold all the different sites apache will host

sudo mkdir /var/www/sites

Modify the group for /var/www

sudo chgrp -R admin /var/www/sites

Allow the group to write

sudo chmod -R g+w /var/www/sites

Install & Configure vnStat (optional) (unconfirmed)

Install vnstat from our yum repository

yum install vnstat.i386

TODO

TODO

Install PHP (unconfirmed)

Install php 5.3 from our yum repository

sudo yum install php53.i386 php53-gd.i386 php53-mbstring.i386 php53-mysql.i386

Restart apache

sudo /etc/init.d/httpd restart

Configure PHP (confirmed)

Modify the php configuration file

sudo vi /etc/php.ini

For future changes

```
...
```

Modify the php apache configuration file

sudo vi /etc/httpd/conf.d/php.conf

Ensure expose_php is set to off

```
expose_php = Off
```

Restart apache

sudo /etc/init.d/httpd restart

Install Java JDK (confirmed)

Download the java jdk binary rpm

wget http://download.oracle.com/otn-pub/java/jdk/6u29-b11/jdk-6u29-linux-i586-rpm.bin -0 jdk_6u29b11.bin

Give the file execute privileges

sudo chmod a+x jdk_6u29b11.bin

Execute the binary rpm

sudo ./jdk_6u29b11.bin

Clean up our binary rpm

rm -f jdk_6u29b11.bin

Clean up our unpacked rpm leftovers

rm -f sun-javadb-*.rpm
rm -f jdk-*.rpm

Modify the environment variable for all users

sudo vi /etc/profile

Add JAVA_HOME to the end of the file (NOTE: You must log out then back in to get this environment variable in your session)

```
export JAVA_HOME=/usr/java/latest
```

Install Tomcat (confirmed)

Download the latest tomcat version

wget http://mirror.metrocast.net/apache/tomcat/tomcat-6/v6.0.33/bin/apache-tomcat-6.0.33.tar.gz

Explode the tar

tar -xzf apache-tomcat-6.0.33.tar.gz

Remove the original tar

rm apache-tomcat-6.0.33.tar.gz

Create the folder where tomcat will live

sudo mkdir /etc/tomcat

Dump our exploded files into the tomcat directory

sudo mv apache-tomcat-6.0.33/* /etc/tomcat/

Clean up our exploded files

rm -rf apache-tomcat-6.0.33

Modify the environment variable for all users

sudo vi /etc/profile

Add CATALINA_HOME to the end of the file (NOTE: You must log out then back in to get this environment variable in your session)

```
export CATALINA_HOME=/etc/tomcat
```

Create tomcat system group

sudo /usr/sbin/groupadd -r tomcat

Create tomcat system user

sudo /usr/sbin/adduser -n -r -s /sbin/nologin -d /etc/tomcat -g tomcat -c Tomcat tomcat

Make tomcat the owner & group for /etc/tomcat & all of its files (TODO - Is this necessary?)

sudo chown -R tomcat /etc/tomcat
sudo chgrp -R tomcat /etc/tomcat

Configure Tomcat (confirmed)

Since we run multiple tomcat servers, it does not make sense to have a base implementation of tomcat. We need to move our tomcat server implementation into a different folder so it can support multiple servers.

Create our new folder that will contain each of our tomcat servers

sudo mkdir /var/tomcat

Create a default server that we will move the base implementation of tomcat into

sudo mkdir /var/tomcat/default

Move the base implementation into the new folder (NOTE: You can simply delete the base implementation folders if you wont need them as a starting point for future server implementations)

sudo mv /etc/tomcat/conf /var/tomcat/default/conf
sudo mv /etc/tomcat/logs /var/tomcat/default/logs
sudo mv /etc/tomcat/temp /var/tomcat/default/temp
sudo mv /etc/tomcat/webapps /var/tomcat/default/webapps
sudo mv /etc/tomcat/work /var/tomcat/default/work

Make tomcat the owner & group for /var/tomcat & all of its files (TODO - Is this necessary? I think it is for tomcat to be able to startup)

sudo chown -R tomcat /var/tomcat
sudo chgrp -R tomcat /var/tomcat

In order to test your tomcat implementation, run the following commands

cd /etc/tomcat/bin
export CATALINA_BASE=/var/tomcat/default
sudo -u tomcat env CATALINA_BASE=$CATALINA_BASE ./startup.sh
lynx http://localhost:8080
sudo -u tomcat env CATALINA_BASE=$CATALINA_BASE ./shutdown.sh

Install GIT (confirmed)

Install git from the yum repository

sudo yum install git-core

Or Alternatively

wget http://pkgs.repoforge.org/git/perl-Git-1.7.6.4-1.el5.rf.i386.rpm
wget http://pkgs.repoforge.org/git/git-1.7.6.4-1.el5.rf.i386.rpm
sudo rpm -i perl-Git-1.7.6.4-1.el5.rf.i386.rpm git-1.7.6.4-1.el5.rf.i386.rpm
rm git-1.7.6.4-1.el5.rf.i386.rpm
rm perl-Git-1.7.6.4-1.el5.rf.i386.rpm

Configure GIT (confirmed)

Make the folder which git repositories will live in

sudo mkdir /var/git

Create git system user

sudo /usr/sbin/adduser -n -r -s /sbin/nologin -d /var/git -g git -c GIT git

Create git system group

sudo /usr/sbin/groupadd -r git

Make this directory owned by the ‘git’ user, and group ‘gituser’

sudo chown git /var/git
sudo chgrp gituser /var/git

We want to restrict repository creation in this folder to be limited to just the ‘git’ user. But we want the ‘gituser’ group to be able to read. And we want everyone else to not have any access to any of the contents in this directory & its subdirectories

sudo chmod 0750 /var/git

Any future directories made in this folder should inherit the group from /var/git

sudo chmod g+s /var/git

Make a default project (as our git user)

sudo -u git mkdir /var/git/default

Initialize a new (bare) repository here (as our git user), the shared flag ensures that the folder group is preserved during git file creations

sudo -u git git init --bare --shared=group /var/git/default

Add our group to the git repository folder permissions

sudo chgrp -R gituser /var/git/default/

TODO - When a file is checked in, /var/git/default/objects has the new file, it has the correct group name and file permissions**, BUT** the file owner is git_ksimons, **NOT** git. A perl hook script may be in order.

Add GIT users (confirmed)

Create our user groups

sudo /usr/sbin/groupadd gituser

Allow this group to have permission to /var/git

sudo chgrp -R gituser /var/git

Create our new users

sudo /usr/sbin/adduser -n -s /usr/bin/git-shell -d /var/git -g gituser git_ksimons

Create passwords for our users

sudo passwd git_ksimons

Allow our gituser group to SSH in

sudo vi /etc/ssh/sshd_config

Add our group to the white list (at the end of the file)

```
AllowGroups <EXISTING USERS> gituser
```

Tell ssh to reload its config file

sudo /etc/rc.d/init.d/sshd reload

Install Jenkins (confirmed)

Download the jenkins RPM

wget http://pkg.jenkins-ci.org/redhat/RPMS/noarch/jenkins-1.435-1.1.noarch.rpm

Install the RPM

sudo rpm -i jenkins-1.435-1.1.noarch.rpm

Clean up the files

rm jenkins-1.435-1.1.noarch.rpm

Configure Jenkins (confirmed)

Modify jenkins configuration file

sudo vi /etc/sysconfig/jenkins

Turn off the HTTP server by replacing the JENKINS_PORT

```
JENKINS_PORT=”-1”
```

Modify the AJP Port since 8009 is tomcats default AJP port

```
JENKINS_AJP_PORT=”9009”
```

Restart jenkins

sudo /etc/init.d/jenkins restart

Create a directory to hold our ssl certificates

sudo mkdir /etc/httpd/conf/certs

Make the apache user the owner of this directory

sudo chown apache /etc/httpd/conf/certs

Do not let anyone besides apache access this directory

sudo chmod 0700 /etc/httpd/conf/certs

Create a private key

sudo openssl genrsa -des3 -out /etc/httpd/conf/certs/jenkins.key 1024

Create our certificate signing request

sudo openssl req -new -key /etc/httpd/conf/certs/jenkins.key -out /etc/httpd/conf/certs/jenkins.csr

Sign our CSR

sudo openssl x509 -req -days 365 -in /etc/httpd/conf/certs/jenkins.csr -signkey /etc/httpd/conf/certs/jenkins.key -out /etc/httpd/conf/certs/jenkins.crt

Make a copy of our key before we remove its passphrase

sudo cp /etc/httpd/conf/certs/jenkins.key /etc/httpd/conf/certs/jenkins.key.secure

Remove the passphrase from our certificate (so apache can use it)

sudo openssl rsa -in /etc/httpd/conf/certs/jenkins.key.secure -out /etc/httpd/conf/certs/

jenkins.key

Add a vhost configuration to apache (NOTE: This must be in the format vhost_*.conf because vhost.conf needs to be loaded first)

sudo vi /etc/httpd/conf.d/vhost_jenkins.conf

The file should contain

```
<VirtualHost *:443>
        SSLEngine on
        SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP
        SSLCertificateFile conf/certs/jenkins.codotos.com.crt
        SSLCertificateKeyFile conf/certs/jenkins.codotos.com.key
        ServerName jenkins.codotos.com
        ServerAlias jenkins.codotos.com
        Include conf.d/mod_jk/jenkins.conf
        ServerAdmin webmaster@jenkins.codotos.com
        ErrorLog logs/jenkins_error_log
        CustomLog logs/jenkins_log common
</VirtualHost>
```

Create a folder to hold our mod_jk configuration files

sudo mkdir /etc/httpd/conf.d/mod_jk

Tell apache all requests should be sent to the mod_jk worker (and handled by jenkins)

sudo vi /etc/httpd/conf.d/mod_jk/jenkins.conf

The file should contain

```
JkMount /* jenkins
```

Create a mod_jk worker to communicate between apache & our tomcat server

sudo vi /etc/httpd/conf.d/workers.properties

The file should contain the following

```
# Define the workers
worker.list=jenkins

# Set properties for testtomcat worker
worker.jenkins.type=ajp13
worker.jenkins.host=localhost
worker.jenkins.port=9009
worker.jenkins.lbfactor=50
worker.jenkins.socket_keepalive=1
```

Restart apache

sudo /etc/init.d/httpd restart

Test Application (Apache & PHP) (confirmed)

Create a folder to represent our site

mkdir /var/www/sites/testphp

Make the required folders for this site (TODO - Permissions are weird here)

mkdir /var/www/sites/testphp/logs
mkdir /var/www/sites/testphp/html

Create our hello world application

vi /var/www/sites/testphp/html/index.php

The file should contain

```
<?php

echo “Hello World.”;

?>
```

Add a vhost configuration to apache (NOTE: This must be in the format vhost_*.conf because vhost.conf needs to be loaded first)

sudo vi /etc/httpd/conf.d/vhost_testphp.conf

The file should contain

```
<VirtualHost *:80>
        ServerName testphp.com
        ServerAlias testphp.com www.testphp.com
        ServerAdmin webmaster@testphp.com
        DocumentRoot /var/www/sites/testphp/html
        # Allow .htaccess files
        <Directory /var/www/sites/testphp/html>
                AllowOverride All
        </Directory>
        ErrorLog /var/www/sites/testphp/logs/error_log
        CustomLog /var/www/sites/testphp/logs/access_log common
</VirtualHost>
```

Restart apache

sudo /etc/init.d/httpd restart

Modify your local hosts file to point the testphp.com & www.testphp.com to the server

Go to http://www.testphp.com, you should see:
“Hello World”

Test Application (Apache & Tomcat) (confirmed)

Create a folder to represent our site

mkdir /var/www/sites/testtomcat

Make the required folders for this site (TODO - Permissions are weird here)

mkdir /var/www/sites/testtomcat/logs
mkdir /var/www/sites/testtomcat/html

Create one part of our hello world application

vi /var/www/sites/testphp/html/index.html

The file should contain

```
Hello World. I am from Apache.
```

Add a vhost configuration to apache (NOTE: This must be in the format vhost_*.conf because vhost.conf needs to be loaded first)

sudo vi /etc/httpd/conf.d/vhost_testtomcat.conf

The file should contain

```
<VirtualHost *:80>
        ServerName testtomcat.com
        ServerAlias testtomcat.com www.testtomcat.com
        Include conf.d/mod_jk/testtomcat.conf
        ServerAdmin webmaster@testtomcat.com
        DocumentRoot /var/www/sites/testtomcat/html
        # Allow .htaccess files
        <Directory /var/www/sites/testtomcat/html>
                AllowOverride All
        </Directory>
        ErrorLog /var/www/sites/testtomcat/logs/error_log
        CustomLog /var/www/sites/testtomcat/logs/access_log common
</VirtualHost>
```

Create a folder to hold our mod_jk configuration files

sudo mkdir /etc/httpd/conf.d/mod_jk

Tell apache which requests should be sent to the mod_jk worker (and handled by tomcat)

sudo vi /etc/httpd/conf.d/mod_jk/testtomcat.conf

The file should contain

```
JkMount index2.html testtomcat
```

Create a mod_jk worker to communicate between apache & our tomcat server

sudo vi /etc/httpd/conf.d/workers.properties

The file should contain the following

```
# Define the workers
worker.list=testtomcat

# Set properties for testtomcat worker
worker.testtomcat.type=ajp13
worker.testtomcat.host=localhost
worker.testtomcat.port=8009
worker.testtomcat.lbfactor=50
worker.testtomcat.socket_keepalive=1
```

Restart apache

sudo /etc/init.d/httpd restart

Create our new tomcat server folder

sudo -u tomcat mkdir /var/tomcat/testtomcat

Copy the default server implementation to our server

sudo -u tomcat cp -r /var/tomcat/default/* /var/tomcat/testtomcat/

Create the other part of our hello world application

sudo -u tomcat vi /var/tomcat/testtomcat/webapps/ROOT/index2.html

The file should contain

```
Hello World. I am from Tomcat.
```

Start the tomcat server

cd /etc/tomcat/bin
export CATALINA_BASE=/var/tomcat/testtomcat
sudo -u tomcat env CATALINA_BASE=$CATALINA_BASE ./startup.sh

Modify your local hosts file to point the testtomcat.com & www.testtomcat.com to the server

Go to http://testtomcat.com, you should see:
"Hello World. I am from Apache."

Go to http://testtomcat.com/index2.html, you should see:
"Hello World. I am from Tomcat."

Shut down the server

cd /etc/tomcat/bin
export CATALINA_BASE=/var/tomcat/testtomcat
sudo -u tomcat env CATALINA_BASE=$CATALINA_BASE ./shutdown.sh