Apache Web Server

Seup Base Web Server

Install

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

Firewall

```
firewall-cmd --add-service=http
firewall-cmd --add-service=https
```

Start

```
systemctl enable httpd.service
systemctl start httpd.service
systemctl status httpd.service
```

Add Basic Index Page

```
echo "server1" > /var/www/html/index.html
```

Test

From c1 or host OS.

```
curl s1
curl -k https://s1
```

Both should return server1

Setup Virtual Host

Create Document Folders

```
mkdir -p /srv/{default,test}/www
echo "default" > /srv/default/www/index.html
echo "test" > /srv/test/www/index.html
ls -ZR /srv
restorecon -Rv /srv
ls -ZR /srv
```

Configure

Finding Examples

```
rpm -qil httpd | grep doc
view /usr/share/doc/httpd-2.4.6/httpd-vhosts.conf
```

```
vi /etc/httpd/conf.d/default-vhost.conf

<VirtualHost default:80>

DocumentRoot "/srv/default/www"

ServerName www.example.com

ErrorLog "/var/log/httpd/default-error"

CustomLog "/var/log/httpd/default" common

</VirtualHost>


<Directory /srv/default/www>

Require all granted

</Directory>
```

For test

```
vi /etc/httpd/conf.d/test-vhost.conf

<VirtualHost *:80>

DocumentRoot "/srv/test/www"

ServerName test.example.com

ErrorLog "/var/log/httpd/test-error"

CustomLog "/var/log/httpd/test" common

</VirtualHost>


<Directory /srv/test/www>

Require all granted

</Directory>
```

Check Configuration

```
apachectl configtest
```

Apply Configuration

```
systemctl reload httpd
```

Test

You'll need to have a DNS entry for test and www or an /etc/hosts entry.

```
curl www.example.com
default
curl test.example.com
test
```

Setup Python Web App

Docs

```
rpm -qil mod_wsgi
view /usr/share/doc/mod_wsgi-3.4/README
```

Install

```
yum -y install mod_wsgi
```

Configure

```
mkdir -p /srv/wsgiapp/www
cd /srv/wsgiapp/www
curl -O gw/helloWorld.wsgi
```

Assumes app is available from gw server. If not you can create it manually.

```
def application(environ, start_response):
    status = '200 OK'
    output = b'Hello World! Python is so easy!\n'
```

```
response_headers = [('Content-type', 'text/plain'),
                    ('Content-Length', str(len(output)))]
start_response(status, response_headers)

return [output]
```

Restore tags for SeLinux.

```
restorecon -rv /srv/wsgiapp/
```

Test

```
curl wsgiapp.example.com
Hello World! Python is so easy!
```

Configure PHP Web Application

Install Module

```
yum -y install mod_php
```

Configure

```
mkdir -p /srv/phpapp/www
cd /srv/phpapp/www
curl -o index.php gw/helloWorld.php
```

Assumes app is available from gw server. If not you can create it manually.

```

```

Restore tags for SeLinux.

```
restorecon -rv /srv/phpapp/
```

Test

```
curl phpapp.example.com
Hello World! PHP is so easy!
```

Secure Site

The mod_ssl is part of the secure configuration.

Configure

Create index page

```
mkdir -p /srv/secure/www
echo 'Secure Page!' > /srv/secure/www/index.html
restorecon -rv /srv/secure/
```

Create Server Cert

```
man req

openssl req -x509 -nodes -days 365 -newkey rsa:4096 -keyout server.key -out
server.crt



Country Name: US

State: IL

Locality Name: Freeburg

Organization: Example

Unit: IT

hostname: secure.example.com

email: root@example.com
```

Output is the server certificate (server.crt) and server key ((server.key).

Position the cert and key.

```
cp server.crt /etc/pki/tls/certs/
chmod 600 /etc/pki/tls/certs/server.crt

cp server.key /etc/pki/tls/private/

chmod 400 /etc/pki/tls/private/server.key



Note: If you move the files (mv) instead of copy (cp); then you'll need to run
restorecon restorecon -Rv /etc/pki/tls



Create Virtual Host



Start with similar base as other Virtual machines.  Then use conf.d/ssl.conf as
example of extra parameters for ssl.
```

```
<VirtualHost *:443>
    DocumentRoot "/srv/secure/www"
    ServerName secure.example.com
    ErrorLog "/var/log/httpd/secure-error"
    CustomLog "/var/log/httpd/secure" common
    SSLEngine on
    SSLProtocol all -SSLv2
    SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5:!SEED:!IDEA
    SSLCertificateFile /etc/pki/tls/certs/server.crt
    SSLCertificateKeyFile /etc/pki/tls/private/server.key
</VirtualHost>

<Directory /srv/secure/www>

Require all granted

</Directory>
```

Reload and Test

```
systemctl reload httpd
```

From another computer.

```
curl -k https://secure.example.com
Secure Page!
```

The -k option tells curl to ignore self-signed cert.

I