**Install in Linux (Cent OS):**

1. To check your hostname run: should show your Fully Qualified Domain Name (FQDN).

$ hostname

$ hostname -f

1. Update your system:
2. $ yum update
3. **Install and Configure Apache**

$ yum install httpd

1. Modify httpd.conf with your document root directory to point Apache to your site’s files. Add the <IfModule prefork.c> section below to adjust the resource use settings.
2. $ cp /etc/httpd/conf/httpd.conf ~/httpd.conf.backup
3. /etc/httpd/conf/httpd.conf

DocumentRoot "/var/www/html/example.com/public\_html"

...

<IfModule prefork.c>

StartServers 5

MinSpareServers 20

MaxSpareServers 40

MaxRequestWorkers 256

MaxConnectionsPerChild 5500

</IfModule>

1. These settings can also be added to a separate file. The file must be located in the conf.module.d or conf directories, and must end in .conf, since this is the format of files included in the resulting configuration.
2. Configure Name-based Virtual Hosts: You can choose many ways to set up a virtual host. In this section we recommend and explain one of the easier methods.

Within the conf.d directory create vhost.conf to store your virtual host configurations. The example below is a template for website example.com; change the necessary values for your domain:

/etc/httpd/conf.d/vhost.conf

NameVirtualHost \*:80

<VirtualHost \*:80>

ServerAdmin webmaster@example.com

ServerName example.com

ServerAlias www.example.com

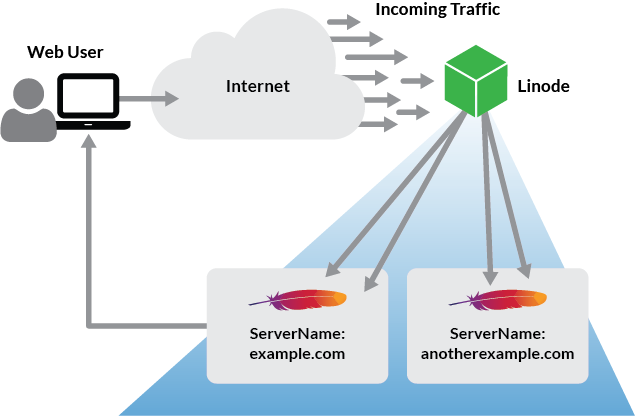
DocumentRoot /var/www/html/example.com/public\_html/

ErrorLog /var/www/html/example.com/logs/error.log

CustomLog /var/www/html/example.com/logs/access.log combined

</VirtualHost>

Additional domains can be added to the vhost.conf file as needed. To add domains, copy the VirtualHost block above and modify its values for each additional virtual host. When new requests come in from the internet, Apache checks which VirtualHost block matches the requested url, and serves the appropriate content:



1. Create the directories referenced above:

$ sudo mkdir -p /var/www/html/example.com/{public\_html,logs}

1. Enable Apache to start at boot, and restart the service for the above changes to take effect:
2. $ systemctl enable httpd.service
3. $ systemctl restart httpd.service
4. You can now visit your domain to test the Apache server. A default Apache page will be visible if no index page is found in your Document Root as declared in /etc/httpd/conf/httpd.conf:
5. Configure firewalld to Allow Web Traffic:

$ sudo firewall-cmd --add-service=http --permanent && sudo firewall-cmd --add-service=https --permanent

$ sudo systemctl restart firewalld

1. Allow the default HTTP and HTTPS port, ports 80 and 443, through firewalld:
2. firewall-cmd --permanent --add-port=80/tcp
3. firewall-cmd --permanent --add-port=443/tcp
4. firewall-cmd –reload

### Prepare the Webapps Directory Structure

The default location for an Apache website on CentOS servers is /var/www/html. This is fine if you do not expect to ever add addtional web applications using Apache’s virtual hosts. Otherwise, it’s best to create your own directory structure.

Before we install Apache and start defining web sites, we need to prepare our webapps directory. Inside of the /srv/webapps directory, we’re going to create separate directories for each of our applications and their logs (see example below), starting with what we’ll call App1.

/srv

|---/webapps

|---/app1

|---/public\_html

|---/logs

1. Create application directorymkdir -p /srv/webapps/app1/public\_html
2. Create application log directorymkdir -p /srv/webapps/app1/logs
3. Change /webapps group ownership to our webadmins.chgrp -Rv webadmins /srv/webapps
4. Set the guid bit recursively for the webapps directory to ensure all new files and directories are owned by the webadmins group. Also, we’re going to grant read/write access to the webadmins group and file owners, and no access to anyone else.chmod 2770 -Rv /srv/webapps
5. With the public\_html directory, we’re going to grant read access to everyone.chmod 2775 -Rv /srv/webapps/app1/public\_html