

# Creating and Enabling an Apache Vhost

- 1) Start by copying the apache default vhost configuration. This will save you some time:
  - a) **`$sudo cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sites-available/[domain].conf`**
- 2) *Optional: Disable the default vhost that we no longer need.*
  - a) **`$sudo a2dissite 000-default`**
- 3) Rename the default webroot and its contained log files:
  - a) **`$sudo mv /var/www/html/domain /var/www/html/[domain]`**
  - b) You can also simply copy your source to a new folder if you'd like
  - c) **`$sudo cp -R /var/www/html/[source] /var/www/html/[destination]`**
- 4) Open your vhost config for editing. You may use vim or nano, as well.
  - a) **`$sudo vi /etc/apache2/sites-available/[domain].conf`**
  - b) Set directory paths correctly. ServerName should be www version, alias will be non-www version.
- 5) Now reload apache, to test configuration
  - a) **`$sudo service apache2 reload`**
    - i) This will fail if Apache is off. To run it: **`$sudo service apache2 start`**
    - ii) You may also see this fail if there is a configuration error. If you get that error, you may be asked to review a log. Review that to find out the mistake you made, then re-test.
- 6) Make sure apache2 is set to run on boot.
  - a) **`$sudo update-rc.d apache2 enable`**
  - b) NOTE: Ubuntu 18+ uses Systemd instead of Sysv, so instead of "service", you may have to use "systemctl"
- 7) Enable your vhost: **`$sudo a2ensite [domain]`**
- 8) Activate your changes: **`$sudo service apache2 reload`**
- 9) Double-check that your local firewall and your hosting provider's firewall both have HTTP and HTTPS ports open (**80 and 443**). Then test access to your new domain.
  - a) You will need to change your DNS to point to your server's address, so you can wait for that to be done or you can edit your computer's hosts file (**ie. /etc/hosts or C:\windows\system32\drivers\etc\hosts**) to point your domain to your server address for testing.