A Fedora based Personal Web Server.

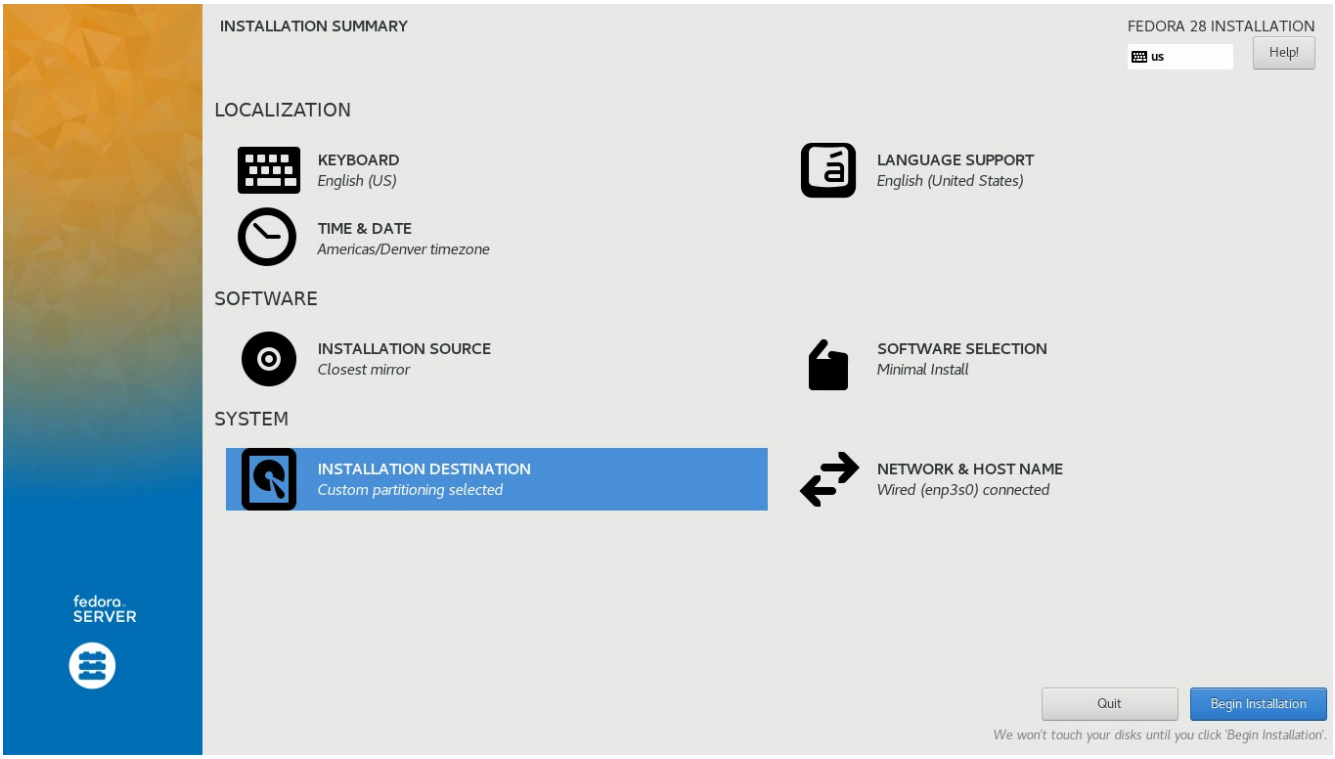
For this to be anything more than just an educational project, you will need to utilize a Dynamic Domain Name Server (DDNS) service. There are several DDNS providers. The one I use is no-ip.com Go to the web site of one of the DDNS providers and do some research. no-ip.com has some good tutorials that will walk you through how a DDNS works. Then decide if a DDNS is something you can and want to do.

This guide installs the Fedora “Minimal” OS on a 60 GB SSD, and uses a high capacity SSD attached to a SATA port for data storage to create a simple Web Server. Having the OS on one physical device (small SSD), having your data on another physical device (large hard drive), and having an external device (large SSD) for backup is the best setup for when something happens. And we all know that eventually something WILL happen. You want to upgrade to Fedora 29, or the OS gets corrupted.. Having the OS by itself on a separate physical device really simplifies that process. If the data hard drive craps out, having a single physical hard drive with a single partition containing nothing but your data on it, and mirroring that data on the external hard drive using rsync makes it easy. Take the SSD out of the external enclosure, put it in the server, edit the /etc/fstab file to change the UUID and you are back in business. Buy a new SSD, put it into the external enclosure, format it, rsync, you're done.

To install the OS on the SSD, a monitor, mouse, and keyboard will need to be connected to the Web Server computer. If ssh is set up, you can do everything remotely and run the web server headless.

Go to fedoraproject.org and download the 64 bit Fedora Server NETINSTALL IMAGE under the OTHER DOWNLOADS.section. Burn the iso to a DVD or USB stick according to Instructions on the Fedora Web Site..

ON THE WEB SERVER COMPUTER.
Configure your computer's BIOS to boot from the DVD/CD optical drive and boot the computer. After selecting a language, click on continue. You should get the INSTALLATION SUMMARY screen

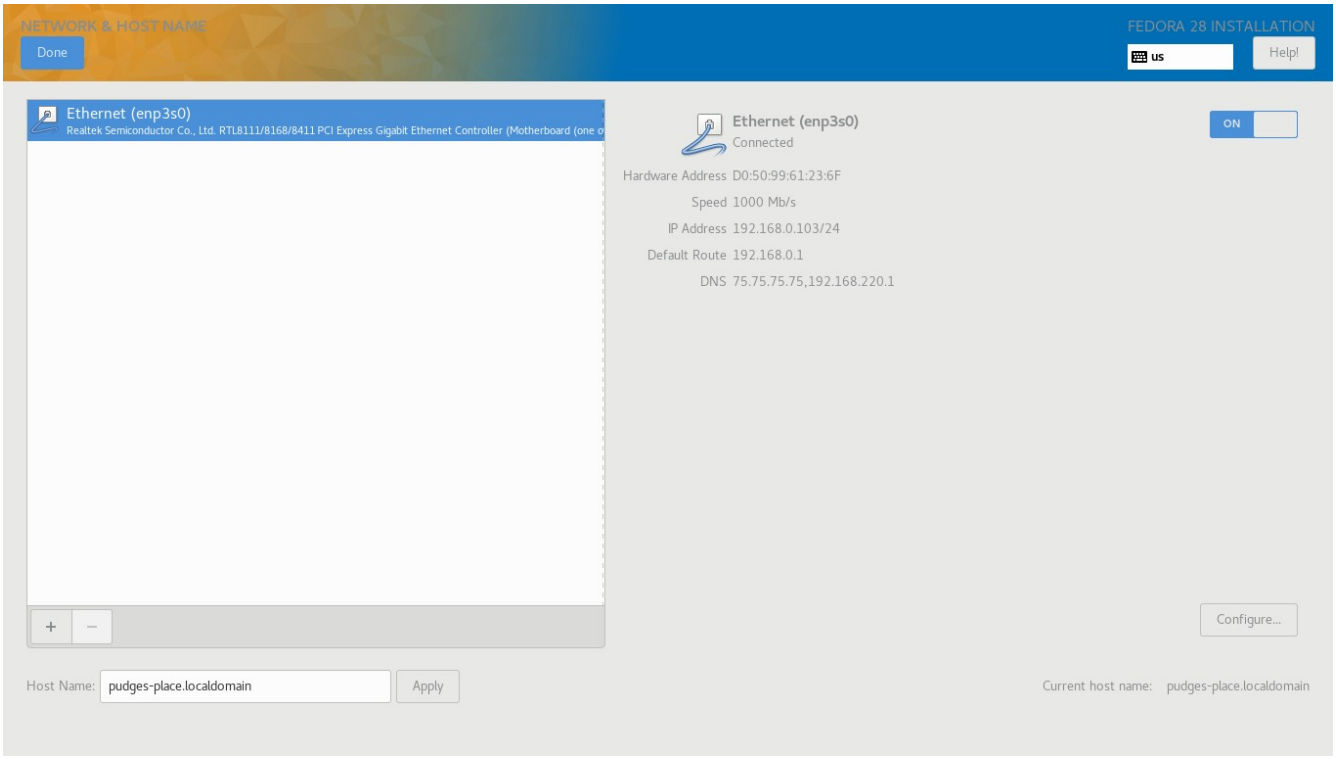


Wait until INSTALLATION SOURCE changes from “Downloading Group Metadata” to “Closest Mirror” and SOFTWARE SELECTION should change to “Fedora Server Edition” before going any further.

Click on KEYBOARD to change keyboard layout if necessary.
Click on LANGUAGE SUPPORT to change language if necessary.
Click on DATE & TIME to set time zone, date, and time if necessary.

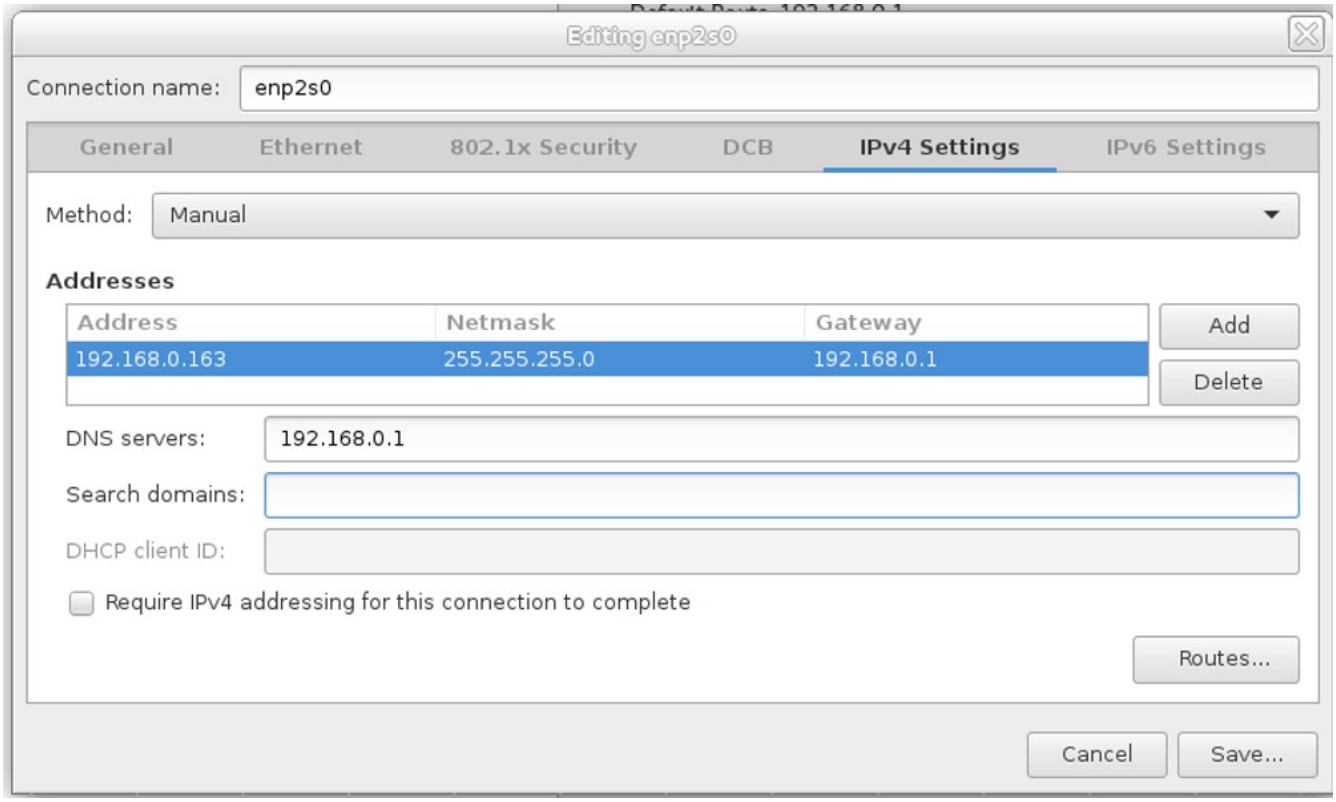
Leave INSTALLATION SOURCE as is: “Closest Mirror”
Click on “SOFTWARE SELECTION” then change to “Minimal Install”

Click on NETWORK & HOSTNAME. The following screen should appear



Go to the lower left to HOSTNAME and change to pudges-place**.**localdomain (or whatever hostname you want) then click on “Apply”

Go to lower right and Click on Configure and you should get the following.



Click on the "IPv4 Settings" Tab. Click on "Method" and change to Manual

I will be using a static IP address of 192.168.0.100 in this guide. Change to what is appropriate in your case. In your static IP address, the first three triads (in my case 192.168.0) are determined by the device your Web Server is connected to. This device is usually a router. It could be something other than 192.168.0 such as 192.168.1 or even 10.0.0 You can set the last triad to whatever you want between 5 and 250 for most routers.

Click on the Add button, then click in the Address entry field.
Enter the desired static IP address, such as 192.168.0.100

Click "Tab" and in the Netmask entry field enter /24

Click in the Gateway entry field and enter 192.168.0.1 (change as appropriate)

Click in the DNS servers entry field and enter 192.168.0.1 (change as appropriate)

The SAVE button should not be grayed out anymore, click on Save.
When you get to the NETWORK CONFIGURATION screen, click on Done and you will return to the INSTALLATION SUMMARY screen.

We need to partition and format the SSD or Hard Drive for the OS. There are two options.
1. Let the Anaconda installer do it for you, which will utilize LVM
2. Do a custom partitioning scheme, without LVM.

For my simple home server, I do not need a LVM (Logical Volume Manager). LVM is great for enterprise servers that need to be scalable, etc. For my simple web server, LVM is just another software layer affecting the filesystem. Not using LVM just means one less thing to go wrong. On my 60 GB SSD, I do custom partitioning using four standard partitions.

/boot boot partition
/ root partition
/home home partition
swap swap partition

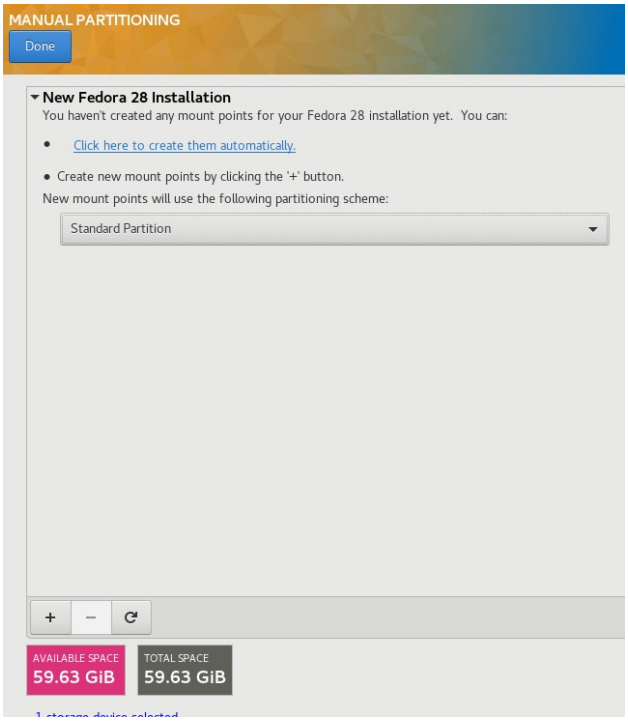
Letting Anaconda installer set up the partitioning with LVM is simple enough. If help is necessary, there are many Fedora installation tutorials on the internet for assistance.

If you want to consider doing a Custom or Manual partitioning, go here:

https://docs-old.fedoraproject.org/en-US/Fedora/26/html/Installation_Guide/sect-installation-gui-manual-partitioning.html

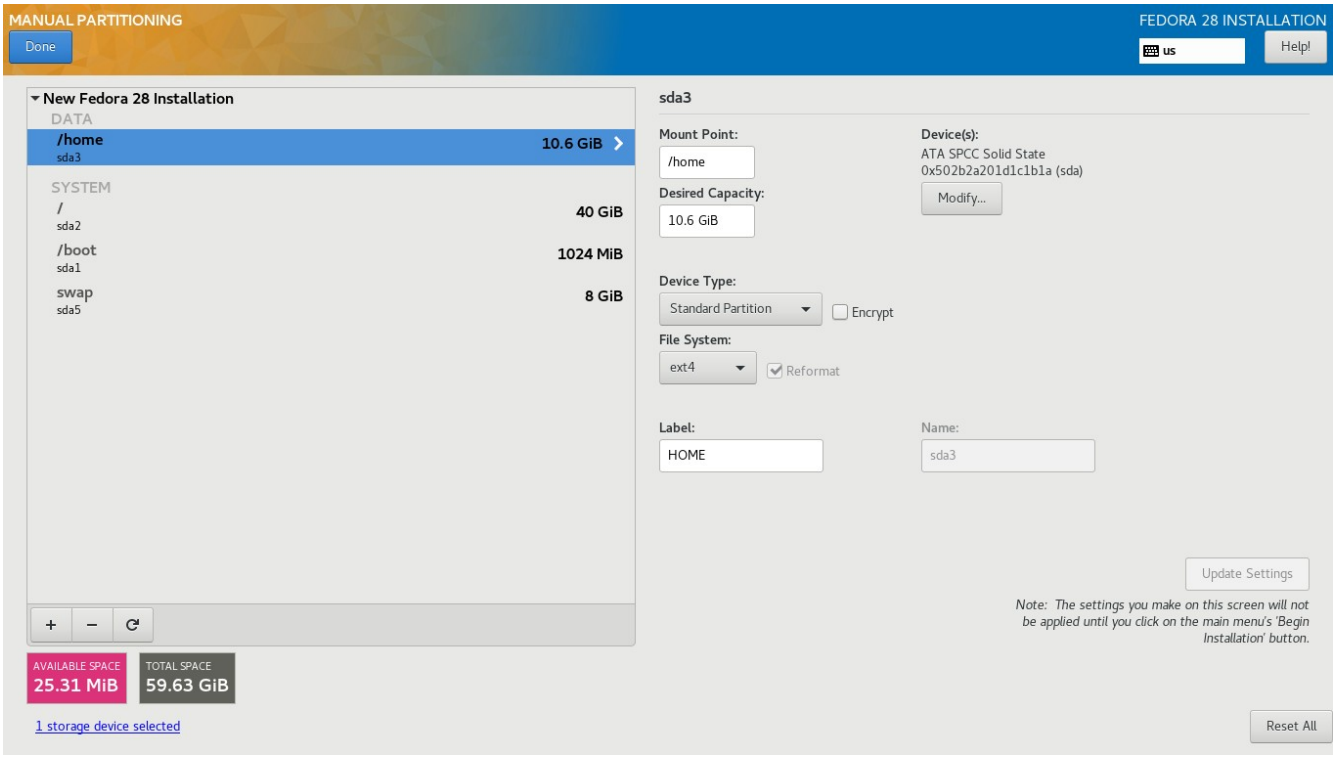
Here is how I manually partition a SSD for the OS.

Click on the INSTALLATION DESTINATION button, select the drive to be partitioned, click on “custom”, then click on “Done”.



Existing partitions will be listed under the drop down menu. Highlight a partition then delete it by using the minus icon until all partitions are deleted. Change the drop down menu to “Standard Partition”.

When the left side of your Manual Partition window looks like this, start adding partitions using the plus icon. Add the /boot partition first so it ends up being the first partition, such as sda1 or sdb1. When all partitions are added, it should look something like below.



Double check the information on the right for each partition. Check Mount Point, Desired Capacity, Device type (Standard Partition), Format type with Reformat checked, and give each partition an appropriate Label.

When finished, click on “Done” then click on Accept Changes and you should return to the “Installation Summary window”.

Once you are happy with your configurations, and there are no Orange Triangle warnings remaining, click on “Begin Installation”.

While Fedora is being installed, click on “ROOT PASSWORD” and set the root password. Click on Done when finished.

Next, click on USER CREATION and in the FULL NAME entry field enter your desired username In the USERNAME field and enter a password. Click on Done when finished.

After installation, be sure the installation USB device is removed. Enter BIOS and set the SSD as the primary boot device if necessary. Boot the computer. You should get a login screen similar to this:

```
Fedora 28 (Twenty Eight)
Kernel 4.18.4-200.fc28.x86_64 on an x86_64 (tty1)

pudges-place login:
```

Log in as root by typing in root for the username, and then enter your root password.

Check that installation set your static IP address and hostname correctly.

```
# ip addr
```

```
<snip>
  2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
   group default qlen 1000
   link/ether d0:50:99:71:1d:6e brd ff:ff:ff:ff:ff:ff
   inet 192.168.0.100/24 brd 192.168.0.255 scope global dynamic enp3s0
<snip>
```

Look for the above snippet. In this case enp3s0 (in red) is the name of the Ethernet interface. Your interface name may be different.

inet should be the Static IP address set during install. If not use vi or nano to edit ifcfg-enp3s0

```
# vi /etc/sysconfig/network-scripts/ifcfg-enp3s0  (change if necessary)
```

then change the ip address.

Note: To change IP address, Gateway, DNS, etc.

```
# vi /etc/sysconfig/network-scripts/ifcfg-enp3s0
```

(substitute your ethernet device name for enp3s0)
Adjust these four lines as appropriate.

```
IPADDR=192.168.0.100
PREFIX=24
GATEWAY=192.168.0.1
DNS1=192.168.0.1
```

Check the hostname for your server

```
# hostnamectl status
```

If the static hostname isn't as set during the install, execute the following

```
# hostnamectl set-hostname pudges-place.localdomain --static
# hostnamectl status
```

SET UP THE FIREWALL

The “Minimal Install” has the firewall set up with the “public” zone as default.
Fedora has added two custom zones, FedoraServer and FedoraWorkstation. The FedoraServer zone will work well for our web server. There is not a space between the double hyphens before each command and/or option. However, without the space, it looks like a single hyphen in this document’s text, so “hyphen space hyphen” was used in place of “hyphen hyphen”.

```
# firewall-cmd  - -get-zones      (list of all supported zones)
  FedoraServer FedoraWorkstation block dmz drop external home internal public trusted work
# firewall-cmd  - -get-default-zone
  public
# firewall-cmd  - -set-default-zone=FedoraServer
  success
# firewall-cmd  - -reload
 # firewall-cmd  - -get-default-zone
  FedoraServer
# firewall-cmd  - -get-active-zones    (should only list FedoraServer)
  FedoraServer
      interfaces: enp3s0
# firewall-cmd   - -permanent  - -zone=FedoraServer  - -remove.service=cockpit
  success
# firewall-cmd  - -reload
 # firewall-cmd  - -list-all     (lists info for the default zone)
   FedoraServer (active)              (These two lines only are of interest)
      services: ssh dhcpv6-client
# firewall-cmd   - -permanent  - -zone=FedoraServer  - -remove-service=ssh
  success
# firewall-cmd   - -permanent  - -zone=FedoraServer  - -add-service=http
  success
# firewall-cmd   - -permanent  - -zone=FedoraServer  - -add-service=https
  success
# firewall-cmd   - -reload
 # firewall-cmd   - -list-all
  FedoraServer (active)
      services:  dhcpv6-client  http  https
# reboot            (reboot to see if new settings hold)
# firewall-cmd  - -get-active-zones
  FedoraServer
      interfaces: enp3s0
# firewall-cmd   - -get-default-zone
  FedoraSever
# firewall-cmd  - -list-all
  FedoraServer (active)
      services: dhcpv6-client  http  https
```

REMOVE SSH

The Minimal Install comes with SSH installed. On a personal LAN server/NAS using ssh to administer the server is appropriate. However, I do not use ssh or Cockpit on a WAN web server for security reasons. Eliminating ssh means one less way someone can hack into your web server. The drawback to no ssh is that a monitor (in my case a cheap yard sale monitor) and a keyboard have to be dedicated to the web server. You can't run a server headless without SSH or Cockpit. I happen to think the drawbacks are worth it. If you want to set up ssh, there are plenty of guides on the internet. If you want to use Cockpit, during installation, select Fedora Server Edition instead of Minimal Install. To get rid of ssh do the following:

```
# systemctl disable sshd.service    (prevent systemd from starting ssh)
# dnf remove openssh
# reboot
```

Since the ssh service was removed from the firewall in the previous section, if you want to use ssh add the ssh service back into the firewall.

```
# firewall-cmd  - -permanent  - -zone=FedoraServer  - -add-service=ssh
  success
```

SET UP THE DATA HARD DRIVE

set up /server & /serverbkup mount points

```
# cd /   (Change to root directory)
# mkdir /server  /serverbkup
# chmod 775 /server  /serverbkup
```

You should now have something similar to this snippet

```
# ll
total 2
drwxrwxr-x.  46 root root 4096 Aug 15 22:15 server
drwxrwxr-x.   2 root root    6    Aug 19 16:29 serverbkup
```

The /server and /serverbkup are directories used for mounting hard drive partitions. You should never put any files or subdirectories in either one of these reserved directories.

```
# poweroff     (power off the computer)
```

Now determine how much DATA storage capacity you need and obtain a SSD or Hard Drive. With the server computer powered down, connect the SSD or Hard Drive to the SATA 2 port on the server's motherboard. Then boot the server computer up. Log in as "root"

blkid

```
/dev/sda1: LABEL="BOOT" UUID="680ccd34-6234-453a-be44-92ec9b2f0262" TYPE="ext4"
/dev/sda2: LABEL="SWAP" UUID="5b247ec7-c3f3-4895-a651-1736e2013beb"
TYPE="swap"
/dev/sda3: LABEL="ROOT" UUID="ea93b6f3-cc33-4f8f-a2b6-d5877d35fa5e" TYPE="ext4"
/dev/sda5: LABEL="HOME" UUID="f1b1e022-56b4-4749-b553-b0f051691f58" TYPE="ext4"
/dev/sdb1: LABEL="####" UUID="f0f0a38a-11c2-4608-adc7-68d88c1863b6" TYPE="####"
```

The output of the blkid command gives us clues as to the dev assignment of the drives. The 4 partition labels listed for /dev/sda shows our four partitions that are obviously the drive that has our Operating System on it. In this case, /de*v*/sda. So /de*v*/sdb is our data drive we just connected. The output for /dev/sdb may vary depending on whether that SSD has ever been formatted, such as a brand new SSD Drive. We have determined in this case that the newly added Data Drive is /dev/sdb. So let's partition and format /dev/sdb

dd if=/dev/zero of=/dev/sdb bs=1M count=8 (zeros out the GPT or MBR)
fdisk /dev/sdb

a. Type in o (the lower case letter o not zero) to clear out any left over partitions
b. Type p to list partitions, there s/b no partitions left
c. Type in n for new partition, then p for a primary partition, then 1 for the first partition, then 2048 for the first sector, then press ENTER to accept the default last sector.
d. Type in w to write the partition and exit.

We just made the entire disk a single primary partition. Now to format this partition.

mkfs.ext4 /dev/sdb1 -L DATA (the -L option sets the volume label for the partition)

We just formatted the single partition to ext4. Part of that formatting includes assigning a new UUID number to the drive. Perform another blkid to find it's new UUID

blkid

```
/dev/sda1: LABEL="BOOT" UUID="680ccd34-6234-453a-be44-92ec9b2f0262" TYPE="ext4"
/dev/sda2: LABEL="SWAP" UUID="5b247ec7-c3f3-4895-a651-1736e2013beb"
TYPE="swap"
/dev/sda3: LABEL="ROOT" UUID="ea93b6f3-cc33-4f8f-a2b6-d5877d35fa5e" TYPE="ext4"
/dev/sda5: LABEL="HOME" UUID="f1b1e022-56b4-4749-b553-b0f051691f58" TYPE="ext4"
/dev/sdb1: LABEL="DATA" UUID="554ac9cc-8df5-44df-b1bd-dc489cc1f9c7" TYPE="ext4"
```

Write down the UUID of /dev/sdb1
Note in the above how judicious use of Labels during partitioning pays off in a blkid command.

Set up to mount Data SSD to mount point /server at boot up.

Your /etc/fstab file should look something like this, except I omitted the PARTUUID for brevity.

cat /etc/fstab (This lists the contents of fstab, I omitted comments)

```
UUID=ea93b6f3-cc33-4f8f-a2b6-d5877d35fa5e      /          ext4    defaults        1 1
UUID=680ccd34-6234-453a-be44-92ec9b2f0262   /boot    ext4    defaults        1 2
UUID=f1b1e022-56b4-4749-b553-b0f051691f58    /home   ext4    defaults        1 2
UUID=5b247ec7-c3f3-4895-a651-1736e2013beb   swap    swap  defaults        0 0
```

Add the following line to /etc/fstab using the new UUID
vi /etc/fstab

UUID= YourNewUUID /server ext4 defaults 1 2

Be sure you specify the format you used when partitioning the Data Drive. If unsure, enter
blkid to list the formats of connected drives.

Your /etc/fstab file should now look something like this now:
cat /etc/fstab

```
UUID=ea93b6f3-cc33-4f8f-a2b6-d5877d35fa5e /                       ext4    defaults        1 1
UUID=680ccd34-6234-453a-be44-92ec9b2f0262 /boot                   ext4    defaults        1 2
UUID=f1b1e022-56b4-4749-b553-b0f051691f58 /home                   ext4    defaults        1 2
UUID=5b247ec7-c3f3-4895-a651-1736e2013beb swap                    swap    defaults        0 0
UUID=554ac9cc-8df5-44df-b1bd-dc489cc1f9c7  /server                ext4    defaults        1 2
```

Note the additional line at the end of the above listing, that is the Data Drive
Reboot the computer

After reboot, the SATA Data Drive should be mounted as /server.
Login as root.
ll /server

```
drwx------. 2 root root 16384 Feb  7 14:31 lost+found
```

should list a directory created at formatting named lost+found.

Installing NGINX

Make sure Fedora is up-to-date

dnf update
dnf install nginx

systemctl enable nginx**.**service

#systemctl start nginx**.**service

In the “Set up the Firewall” section, ports 80 (http) and 443 (https) were opened.
If this didn’t get done:

firewall-cmd - -permanent - -zone=FedoraServer - -add-service=http
firewall-cmd - -permanent - -zone=FedoraServer - -add-service=https

The default NGINX root directory is
/usr/share/nginx/html

When a browser is directed to your base URL, this is where NGINX will look for the index.html file that is your front page. We need to change the NGINX root directory

vi /etc/nginx/nginx.conf

change the following line in the “Server” section
  root /usr/share/nginx/html;
to
  # root /usr/share/nginx/html; (# indicates the following is a comment and ignored)
then add the following line
  root /server;
This allows you to “Go Back” by commenting this line and uncommenting the original line.

If the new nginx home data directory is anything but a sub-directory of /usr/share/nginx/, then Selinux must be notified of the new path. Otherwise you’ll get a “403 Forbidden” error. In our case, /server is not a sub-directory of /usr/share/nginx so to notify Selinux:

chcon -Rt httpd_sys_content_t /server/*

Now start setting up your web site in the NGINX root directory, in our case /server, which must contain a HTML file named index.html The /server directory must have “root root” as owners, and 755 (rwxr-xr-x) as permissions. All sub-directories of /server and all files must also have “root root” as owners and 755 as permissions. Use these settings for test purposes.
However, some users don’t think this gives the best security. Once you have a simple web site up and working, play with these settings as you wish. When using your browser to check changes, be sure to refresh your browser window or you may get false results from reading cached info.

As part of installation, the directory /usr/share/nginx/html contains 5 files for testing purposes. Notice the ownership and permissions. These files came from NGINX as part of the installation and is how NGINX recommends setting these.

```
ll /usr/share/nginx/html
total 28
-rwxr-xr-x. 1 root root 3650 May 31 10:47 404.html
-rwxr-xr-x. 1 root root 3693 May 31 10:47 50x.html
-rwxr-xr-x. 1 root root 4566 Aug 15 00:28 index.html
-rwxr-xr-x. 1 root root  368 May 31 10:47 nginx-logo.png
-rwxr-xr-x. 1 root root 2811 May 31 10:47 poweredby.png
```

The index.html file contains a test home page. 404.html and 50x.html are test error codes, and the .png files are image files of nginx and fedora logos. You can cp (don't use mv) these files to /server for testing. Be sure the ownership and permissions are correct.

Set up your router for either virtual server OR port forwarding to the IP address of your web server port 80 (http and port 443 (https)

In your favorite browser, go to whatismyip.com and find your Cable/DSl Modem's public IP address. Then type in the the IP address port 80 such as

81.281.211.210:80

You should get the test home page as follows

DDNS Services

At this point, you can only externally access your web site via the cable modem's public IP address. Most Internet Providers utilize dynamic IP addresses, meaning they can change IP addresses at their digression. Also you cannot bind your public IP address to a domain name. Remembering and typing in IP addresses is a pain. The answer to this is a Dynamic Domain Name Server. There are several providers for this service. I have used no-ip.com free for a year with no problems. I recently upgraded to enhanced DDNS. Go to no-ip.com or any other Dynamic DNS service, and do some research. set up an account with a domain name.

The way this works is a Dynamic DNS update Client runs in the background and when it senses a change in the cable modems public IP address, the client phones home and provides the new public IP address.

For no-ip.com, Download "Dynamic DNS Update Client for Linux" Ver 2.1.9
to /usr/local/src

gunzip and tar the file

cat the README**.**FIRST file

make install Makefile
Answer questions:
Note: Enter login/email string = username

pudges-place.ddns.net pudgy-duck@comcast.net
user name = p-d password = Jody Mascot = SHS mascot

For the Dynamic DNS Update Client to Phone Home, open this port
firewall-cmd --permanent --zone=FedoraServer –add-port=8245/tcp

The client is /usr/local/bin/noip2
chown root:root /usr/local/bin/noip2
chmod 700 /usr/local/bin/noip2

/usr/local/bin/noip2 -C configure client
/usr/loca/bin/noip2 run the client
/usr/local/bin/noip2 -S display info about running clients
/usr/local/bin/noip2 -K pid Terminate client by pid

noip2 configuration file
/usr/local/etc/no-ip2.conf
chown root:nobody /usr/local/etc/no-ip2.conf
chmod 660 /usr/local/etc/no-ip2.conf

SET UP NOIP2 TO START WITH SYSTEMD

To automatically start the noip2 service at boot up, create a unit file for systemd

vi /etc/systemd/system/noip2.service (Add the following lines)

```
[Unit]
Description=noip2 daemon to detect IP address changes
After=nginx.target

[Service]
ExecStart=/usr/local/bin/noip2
Type=forking

[Install]
WantedBy=multi-user.target
```

Save the vi file, then

chmod 755 /etc/systemd/system/noip2.service

systemctl enable noip2.service

systemctl start noip2.service
reboot

After reboot, see if noip2 is running. Commands to see if a service is running
/usr/local/bin/noip2 -S
systemctl -t service (list running services)
systemctl list-unit-files -t service (lists all installed services)
ps aux | grep noip2 (to get a pid number)

SETUP HTTPS (SSL) SERVER

First, make sure all routers have port forwarding for port 443 and any firewalls are set to pass port 443. Open a browser, and enter https://yourdomain.net or whatever is appropriate for your server's domain name. If you get an indication it's connecting by can't set up encryption due to improper certificate, you are ready for the next step.

dnf update
dnf install certbot-nginx

Certbot will obtain a certificate from Let's Encrypt, install the certificate, and configure the file /etc/nginx/nginx.conf for you. All you have to do is answer some questions.

certbot --nginx (then answer the questions)

You can have your server:
receive http (unencrypted) requests and send them back as http
and receive https (encrypted) requests and send them back as https
OR
divert any http (unencrypted) requests to https (encrypted) and send everything back as https

I think the second choice is better and recommend it, because all newer major browsers are starting to send all requests as https.

Some handy commands"
certbox certificates (check if configuration was correct and list status of certificate)
certbox renew --dry-run (checks if your set up will do a certificate renewal)
certbot renew

If certbot renew fails with "incorrect validation certificate for TLS-SNI-01 challenge" try

certbot renew --preferred-challenges http

Once you have everything working, make a copy of /etc/nginx/nginx.conf and etc/nginix/blockips.conf to a USB thumb drive. This will backup all the configurations you made earlier.

cp -a /etc/nginx/nginx.conf etc/nginx/blockips.conf to a USB thumb drive

Then copy the /etc/letsencryp directory to the USB thumb drive.
cp -av /etc/letsencrypt /USBmountpoint

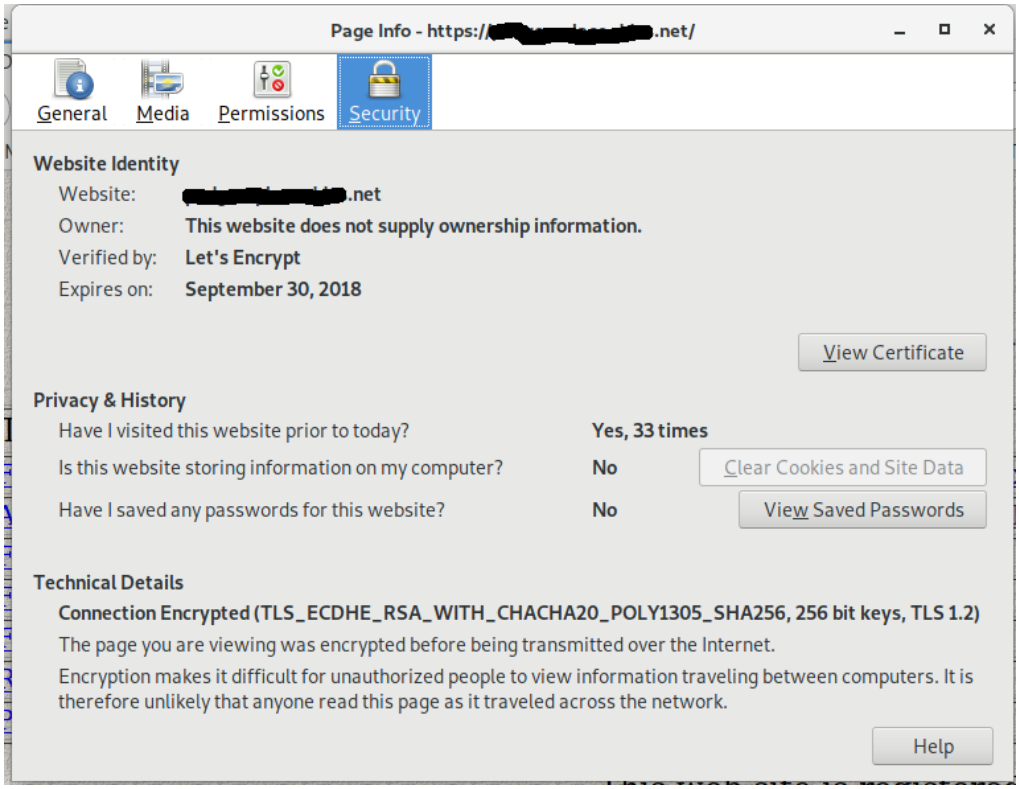
Then if you need to re-install your Web Server's OS, you can recover your certificate.
dnf update
dnf install certbot-nginx (so you will have certbot renew and other commands)
copy your backup blockips.conf and nginx.conf files from USB thumb drive to the computer
cp -av /USBmountpoint/blockips.conf /USBmountpoint/nginx.conf /etc/nginx
copy your letsecnrypt directory from the USB drive to the computer
cp -av /USBmountpoint/letsencrypt /etc

When you did a backup of the letsencrypt directory, 4 important symlinks did not get copied. You have to restore them manually.

cd /etc/letsencrypt/live/yourdomain.net.
pwd (ensure you are in proper directory)
/etc/letsencrypt/live/yourdomain.net
ln -s ../../archive/yourdomain.net/cert1.pem cert.pem
ln -s ../../archive/yourdomian.net/chain1.pem chain.pem
ln -s ../../archive/yourdomain.net/fullchain1.pem fullchain.pem
ln -s ../../archive/yourdomain.net/privkey1.pem privkey.pem
ll (you should see the 4 symlinks and the readme file)
cert.pem -> ../../archive/yourdomain.net/cert1.pem
chain.pem -> ../../archive/yourdomain.net/chain1.pem
fullchain/pem -> ../../archive/yourdomain.net/fullchain1.pem
privkey.pem ../../archive/yourdomain.net/privkey1.pem
README
reboot

After reboot, in your client computer open your browser and test your web site.
If it works, click on the padlock left of the URL and check you expiration date. Renew the certificate if necessary,

Certificates from Let's Encrypt are good for 90 days. AFTER 60 days and BEFORE 90 days renew your certificate. If you want to see the expiration date of your current certificate, open a browser, go to your web site, click on the padlock icon to the left or the URL. You will get a dialog box with the padlock, your domain name, and secure connection at the top. Click on the arrow to the right of this, then click on "more information" at the bottom. You will get:



Click on "View Certificate" for more information. The above has an expiration date of September 30 2018. So on about Sept 3 or 4, get into your server as root and enter:
certbot renew

and it should renew your certificate for another 90 days.
If certbot renew fails with "incorrect validation certificate for TLS-SNI-01 challenge" try

certbot renew --preferred-challenges http

In Fedora, you can setup a systemd timer to automate this process. You might want to look at that process.

I would strongly suggest that you visit the following web sites and read up on installing and to renew certificates.

https://letsencrypt.org/getting-started/

https://certbot.eff.org/

Harden your Web Server

Unfortunately your web site will be probed by hackers, web crawlers, robots, etc. To look at a log of access attempts, add the following lines in the “server {” section of the /etc/nginx/nginx.conf file. Insert lines in Red text.

```
http {

  Yada yada yada
  A bunch of stuff

   server {
      listen       80 default_server;
      listen       [::]:80 default_server;
      server_name  _;
#     root         /usr/share/nginx/html;
      root         /server;

#     set up viewing of access_log
      access_log /var/log/nginx/access_log combined;

      A bunch more stuff
   }

# Settings for a TLS enabled server.
```

Save and close the file. Test the conf file

nginx -t

sample output:
the configuration file /etc/nginx/nginx.conf syntax is ok
configuration file /etc/nginx/nginx.conf test is successful

reboot to reload the new config file or enter # nginx -s reload

Now you can view the contents of the /var/log/nginx/access_log using the CLI cat command
cat /var/log/nginx/access_log

or the CLI less command
less /var/log/nginx/access_log

or you can list updates to an active screen by using the CLI tail command
tail -f /var/log/nginx/access_log

The format of the access_log is:

Remote_Addr - remote_user - [time_local] “request” status body_bytes_sent “http_referer” “http_user_agent

Below are the lines showing what happens when someone types in my domain name and it loads up my home page. It looks like this

104.238.46.64 - - [29/Jul/2017:20:44:18 -0600] "GET / HTTP/1.1" 200 3350 "-" "Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0"
104.238.46.64 - - [29/Jul/2017:20:44:18 -0600] "GET /images/attorney.gif HTTP/1.1" 200 19957 "http://98.245.218.222/" "Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0"
104.238.46.64 - - [29/Jul/2017:20:44:18 -0600] "GET /images/fedora.png HTTP/1.1" 200 3837 "http://98.245.218.222/" "Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0"
104.238.46.64 - - [29/Jul/2017:20:44:18 -0600] "GET /images/nginx-logo.png HTTP/1.1" 200 368 "http://98.245.218.222/" "Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0"
104.238.46.64 - - [29/Jul/2017:20:44:18 -0600] "GET /images/apaper.gif HTTP/1.1" 200 3448 "http://98.245.218.222/" "Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0"
104.238.46.64 - - [29/Jul/2017:20:44:18 -0600] "GET /images/digi.jpg HTTP/1.1" 200 39335 "http://98.245.218.222/" "Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0"

Lets look at the first line
104.238.46.64 is the remote_addr which is the remote users IP address
- - is the remote user in this case undetermined
[29/Jul/2017:20:44:18 -0600] is the time_local which is the date and time of the request
"GET / HTTP/1.1" is the request which in this case is the index.html file for my home page
200 is the status of the request, in this case 200 is no error, other status 403 404 etc.
3350 is the body_bytes_sent, in this case 3,350 bytes or 3.35 Kbytes the size of index.html
“-” is the http_referer, in this case it’s blank
"Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0" Lets break this down

Mozilla/5.0 is the general token that says the browser is Mozilla compatible, and is common to almost every browser today.
(X11; Fedora; Linux x86_64; rv:54.0) is the Operating System of the remote computer. In this case 64 bit Fedora Linux using XORG11 graphics
Gecko/20100101 indicates that the browser is based on the Gecko engine
Firefox/54.0 indicates the browser being used is Firefox Ver. 54

The remaining lines are similar except the request "GET /images/attorney.gif HTTP/1.1" specifies a file, in this case a .GIF file. The file /images/attorney.gif is an image which is part of the home page, which was requested by the index.html file just uploaded. If the “Get” request does not specify a file, then index.html is assumed. Index.html file is a default file. It automatically displays when someone goes to your domain name.

Also, the remaining lines of the access_log file show 104.238.46.64 as the – remote_user -. In this case that is the IP address the Internet Service Provider gave to your cable modem or DSL modem that your computer is hooked to. If a member on a Forum requests a thread that contains an image hosted on your web server, the URL of the Forum will be listed as the – remote_user -

With the above information, it won't take long for you to decipher quite a bit about the accesses to your web server. After a short while you will be able to tell the difference between legitimate accesses and bogus accesses.

Here is a legitimate request by someone on a forum site viewing a thread with a image link for an image hosted on your web server.

204.195.2.246 - - [28/Jul/2017:01:49:03 -0600] "GET /images/prettypicture.jpg HTTP/1.1" 200 2752471 "http://www.someforum.com/forum/category/threadname.html" "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.115 Safari/537.36"

On July 28 2017 someone at IP 204.195.2.246 using Windows 7 64 bit with a Safari web browser was on someforum.com and pulled up threadname.html that included an image link of /images/prettypicture.jpg on this web server and it uploaded the 2752471 byte file successfully.

As far as HTML headers go, here are the references to different Windows Operating Systems

Windows 10 = NT 10 Windows 8.1 = NT 6.3 Windows 8 = NT 6.2 Windows 7 = NT 6.1
Windows Vista = NT 6.0 XP Pro 64 = NT 5.2 Windows XP = NT 5.1
Windows 2000 = NT 5.0 Windows ME = NT 4.9 Windows 98 = NT 4.1

Here are a couple of accesses that are NOT what I want on my static file web server.

177.64.135.26 - - [28/Jul/2017:00:58:32 -0600] "GET /cgi/common.cgi HTTP/1.0" 404 675 "-" "Wget(linux)"
177.64.135.26 - - [28/Jul/2017:00:58:32 -0600] "GET /stssys.htm HTTP/1.0" 404 675 "-" "Wget(linux)"
177.64.135.26 - - [28/Jul/2017:00:58:32 -0600] "GET / HTTP/1.0" 200 3350 "-" "Wget(linux)"
177.64.135.26 - - [28/Jul/2017:00:58:33 -0600] "POST /command.php HTTP/1.0" 404 675 "-" "Wget(linux)"

Here's how I interpret these log entries. On Jul 28 2017 someone at IP 177.64.135.26 using Linux in a Terminal window entered the CLI command Wget and tried to download several different files which resulted in status 404 (file not found) and the file 404.html sent back was 675 Bytes.

My web server is a static file server only. To use the Wget command, I would need to have my server set up as a FTP server, which it is not. So, especially considering the time of day (00:58) I take it as someone sitting at a Linux computer trying to hack into my web server. He got 404 (file not found) on three attempts. The third attempt he asked for my index.html and evidently got it since it had a status of 200 (no errors).

At this point I would blacklist this IP address so ANY attempt to access the web server would result in status 403 (Forbidden).

To block access to your web server by IP address, add the following lines in the “http {” section of the /etc/nginx/nginx.conf file. Insert lines in Red text just before the “server {“ section

```
http {
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';    Yada Yada Yada
    A bunch of stuff

   # block spammers and unwanted visitors by IP address
    include blockips.conf;

    server {
        listen       80 default_server;
        listen       [::]:80 default_server;
        server_name  _;
```

Save and close the file. Use vi to create the file /etc/nginx/blockips.conf
add the IP adresses you want to be blocked as follows.

deny 91.230.47.3;
deny 177.64.135.26;
so forth and so forth

save and close the blockips.conf file, and test the config file

nginx -t

sample output:
the configuration file /etc/nginx/nginx.conf syntax is ok
configuration file /etc/nginx/nginx.conf test is successful

Reload the new configuration
nginx -s reload

ANY attempts to access the web server from these IP addresses would result in status 403 (Forbidden)

91.230.47.3 - - [29/Jul/2017:14:36:43 -0600] "GET / HTTP/1.0" 403 169 "-" "-"

Since this is on a ISP residential account, I don't want web crawlers using a bunch of bandwidth. My ISP may not like that.

157.55.39.133 - - [28/Jul/2017:03:04:52 -0600] "GET /robots.txt HTTP/1.1" 404 675 "-" "Mozilla/5.0 (compatible; bingbot/2.0; +http://www.bing.com/bingbot.htm)"
185.110.132.239 - - [28/Jul/2017:06:37:06 -0600] "GET / HTTP/1.1" 200 3350 "-" "Scanbot"
141.8.144.123 - - [28/Jul/2017:10:29:08 -0600] "GET /robots.txt HTTP/1.1" 404 675 "-" "Mozilla/5.0 (compatible; YandexBot/3.0; +http://yandex.com/bots)"
104.128.144.131 - - [28/Jul/2017:21:54:50 -0600] "GET / HTTP/1.0" 200 3350 "-" "www.probethenet.com scanner"
46.229.164.98 - - [29/Jul/2017:02:36:36 -0600] "GET /robots.txt HTTP/1.1" 404 675 "-" "Mozilla/5.0 (compatible; SemrushBot/1.2~bl; +http://www.semrush.com/bot.html)"

These are access attempts by bots, or webcrawlers trying to probe my web site. They are in order
bingbot from www.bing.com
YandexBot from yandex.com/bots
probethenet scanner from probethenet.com
SemrushBot from semrush.com

There are numerous more such as Scanbot, masscan, google, zgrab, etc. If you were running a commercial site, then bots like google, bing, and etc would be desirable as it would generate more traffic for your site.

93.115.254.200 - - [29/Jul/2017:10:20:05 -0600] "GET /images/gouges.jpg HTTP/1.1" 200 876719 "https://www.google.ro/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.115 Safari/537.36"

There is Google, Google maps, Google images, etc. In the above access, google followed up on probes of my web server, then www.google.ro came in and pirated one of my images, namely /images/gouges.jpg. Which was 8.7 Mbytes. I don't want bots running up my bandwidth by uploading my images, and images are a big target for these bots. Just go to Google, select images in the upper right, then type in "rifle". You instantly get thousands of pictures of rifles, and click on an image, and you can go to the web site the picture came from. How do you think they get these images?

Fortunately, there is a way to block these bots. Edit the /etc/nginx/nginx.conf file, and add add the following lines in the “server {” section of the /etc/nginx/nginx.conf file. Insert lines in Red text. Note that the lines

location / {

}

may be in the nginx.conf file as a default. If so, just add the lines in the middle. Enter keywords of the bot you want separated by the | (pipe or broken bar symbol).

57.55.39.133 - - [28/Jul/2017:03:04:52 -0600] "GET /robots.txt HTTP/1.1" 404 675 "-" "Mozilla/5.0 (compatible; bingbot/2.0; +http://www.bing.com/bingbot.htm)"
185.110.132.239 - - [28/Jul/2017:06:37:06 -0600] "GET / HTTP/1.1" 200 3350 "-" "Scanbot"

Using http_user_agent you can use bing and Scanbot as keywords for the two examples above

```
server {
        listen       80 default_server;
        listen       [::]:80 default_server;
        server_name  _;
  #       root         /usr/share/nginx/html;
          root          /server;
          access_log /var/log/nginx/access_log combined;

        # Load configuration files for the default server block.
        include /etc/nginx/default.d/*.conf;

          location / {
            if ($http_user_agent ~* (bing|Scanbot|Googlebot-Image|Google|Wget||yandex|
Scanbot|probethenet|semrush|masscan|zgrab) )  {
              return 403;
            }
          }

        error_page 404 /404.html;
            location = /40x.html {
        }

        error_page 500 502 503 504 /50x.html;
            location = /50x.html {
        }
    }
```

If any of the keywords appear in the http_user_agent field, it results in status 403 (Forbidden)

Backing up your Web Server Data Files

Remember when we made two directories right off root named /server and /serverbkup? Well /serverbkup is for mounting an external USB hard drive enclosure to backup your Web server.

Ealier, a Gparted Live DVD was used to partition and format a SATA hard drive for our DATA drive. Now we will use the same procedure to partition and format an external USB hard drive for a backup hard drive. Ideally the USB hard drive should be the same size as the SATA Data hard drive. In my case both are 500 GB. We are going to format the USB hard drive exactly like the Data hard drive. One single partition formatted so that it has the same format (ext3 or ext4) as the SATA Data Drive's format. There is a reason for all this. IF the external USB hard drive enclosure has a SATA hard drive inside the enclosure, and IF the SATA Data Drive goes bad, all that needs to be done is take the hard drive out of the USB enclosure and swap it with the defective Data hard drive. Then edit the /etc/fstab file and change the UUID of the old Data Hard drive to the UUID of the backup hard drive. Voila, you are back up and running in about 20 to 30 minutes.

Connect a USB external hard drive to the server computer. For safety, I make sure that the hard drive I intend to partition and format is the ONLY hard drive connected by temporarily unplugging the installed SATA DATA hard drive and the USB thumb drive. During boot up, change BIOS so it boots from the Gparted DVD and partition and format the backup USB hard drive. Power off the computer and the USB hard drive and restore the computer to it's original condition.

I leave the USB external enclosure connected and powered down except when doing a back up. To do a backup:

In the Web Server as root:
dnf install rsync (this only needs to be done the first time to install rsync)
fdisk -l

Determine which devices are being used. Most likely /dev/sda and /dev/sdb
Power up the external USB hard drive
fdisk -l
Determine the new device that appeared after power up. Most likely the additional device will be /dev/sdc so in this example I will use /dev/sdc1

# ll /serverbkup (should list zero files since the external drive is not mounted)
# mount -t ext3 /dev/sdc1 /serverbkup (be sure to use the proper format type ext3 or ext4)
# ll /serverbkup (should list the files proving external drive is mounted)
# su username (preform the backup as user)
$ rsync -av –delete /server/ /serverbkup (the / at the end of /server/ is important)
$ exit (back to root)
# umount /serverbkup
# ll /serverbkup (should list zero files proving the external drive has been unmounted)
Once umounted, power off the external USB hard drive

Addendum

Some useful firewall-cmd commands:
firewall-cmd --get-default-zone (lists the default zone)
firewall-cmd --set-default-zone=public (sets a new default zone)
firewall-cmd --list-all (Lists info for the default zone)
firewall-cmd --list-services (lists active predefined services)
firewall-cmd --list-all –zone=FedoraServer (or other zones such as public)
firewall-cmd --reload (makes any changes active – see permanent and runtime settings)
firewall-cmd --query-service=ssh (list if a service such as ssh is enabled)
firewall-cmd --get-zones (List of all supported zones)
firewall-cmd --state (get state of firewall)
firewall-cmd --get-services (get list of all supported services)
firewall-cmd --list-all-zones (list all zones with the enabled features)
firewall-cmd --get-active-zones
firewall-cmd --permanent --zone=FedoraServer –add-service=http

firewall-cmd --help (the ultimate firewall-cmd)

USER'S NOTES: