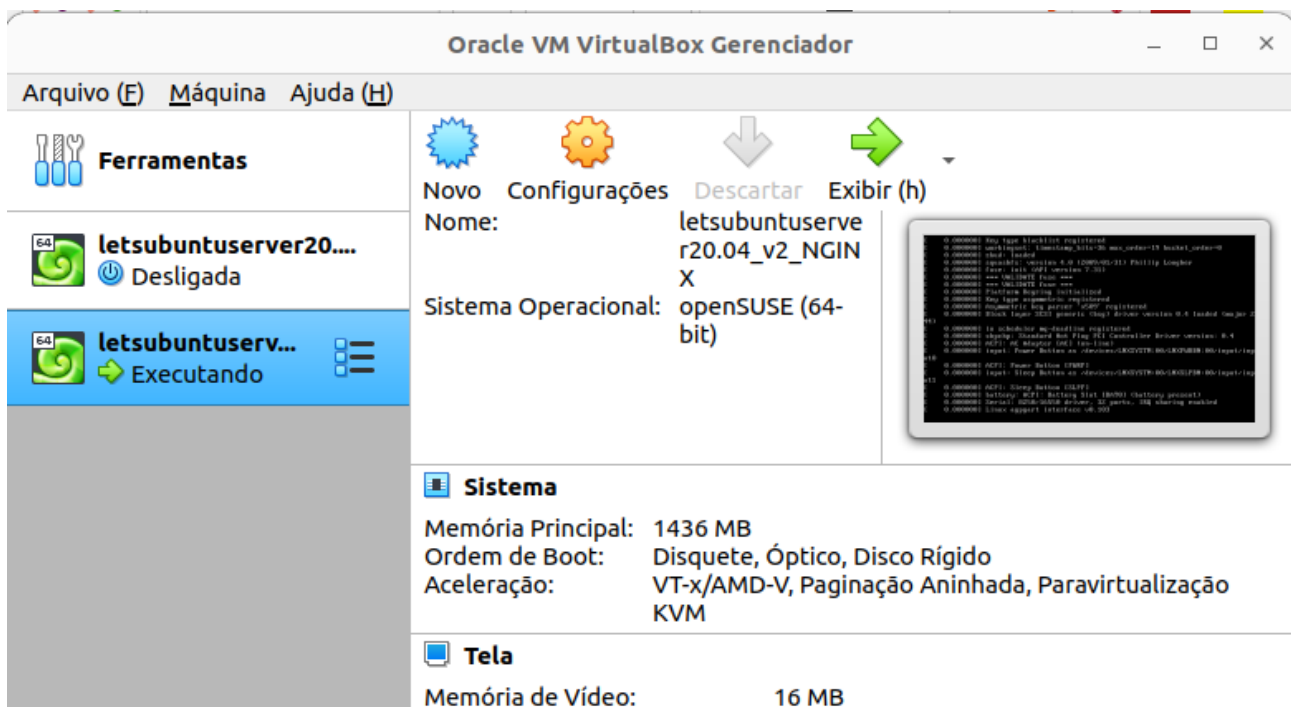


Módulo 09 – Servidores – Atividade 05

Exercícios:

- 1.Desinstale Apache2 instalado em aulas anteriores.
- 2.Instale o Nginx, utilizando o arquivo binário em sua última versão.
- 3.Demonstre os passos que foram realizados para a compilação do Nginx na sua última versão estável.
- 4.Demonstre a geração das chaves pública e privada a serem utilizadas na configuração do suporte a HTTPS.
- 5.Apresente as alterações feitas nos arquivos de configuração do Nginx para suporte a conexões seguras.
- 6.Envie arquivos por meio de FTP 'chrooted' e exiba o resultado obtido.

Q1) Para não desinstalar o Apache2, optei por criar um clone na Virtualbox. Portanto, tem-se duas máquinas virtuais, uma com o Apache2 e outra onde será instalado apenas o NGINX.



Q2) e Q3) Estou utilizando virtualbox no ubuntu.

O primeiro passo é utilizar o seguinte comando no terminal da máquina local:

```
VBoxManage startvm "letsubuntuserver20.04_v2_NGINX" --type headless
```

Então, aplico a conexão ssh para poder utilizar o terminal do sistema operacional para acessar a máquina virtual, através do comando:

```
“ssh letsubuntu@192.168.0.113”
```

O IP acima é o da máquina, lembrando que isso é feito quando se vai nas configurações da máquina virtual, rede, e mudar de NAT para modo Bridge!

```
letsubuntu@letsubuntu: ~  
letonio@letonio-Inspiron-15-3567:~$ VBoxManage startvm "letsubuntuserver20.04_v2_NGINX" --type headless  
Waiting for VM "letsubuntuserver20.04_v2_NGINX" to power on...  
VM "letsubuntuserver20.04_v2_NGINX" has been successfully started.  
letonio@letonio-Inspiron-15-3567:~$ ssh letsubuntu@192.168.0.113  
The authenticity of host '192.168.0.113 (192.168.0.113)' can't be established.  
ED25519 key fingerprint is SHA256:rZ560fUKGQpl8k/2e29fcE28qrqgU+MSlWTwWwAnjBs.  
This host key is known by the following other names/addresses:  
  ~/.ssh/known_hosts:1: [hashed name]  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added '192.168.0.113' (ED25519) to the list of known hosts.  
letsubuntu@192.168.0.113's password:
```

Para fazer a instalação, estarei seguindo o tutorial disponibilizado nas referências, conforme ilustrado abaixo:

https://www.alibabacloud.com/blog/how-to-build-nginx-from-source-on-ubuntu-20-04-lts_597793

Passo 1 – Atualizar e instalar dependências

Já estamos logados no servidor através do comando (ssh letsubuntu@192.168.0.113)

Atualizando o gerenciador de pacotes do Ubuntu através do comando (sudo apt-get update)

Em seguida, instalarei bibliotecas de desenvolvimento juntamente com compiladores de código fonte (sudo apt-get install build-essential libpcre3 libpcre3-dev zlib1g zlib1g-dev libssl-dev libgd-dev libxml2 libxml2-dev uuid-dev)

```
letsubuntu@letsubuntu:~$ sudo apt-get install build-essential libpcre3 libpcre3-dev zlib1g zlib1g-dev libssl-dev libgd-dev libxml2 libxml2-dev uuid-dev  
Reading package lists... Done  
Building dependency tree
```

Passo 2 – Fazer o download do código fonte do NGINX e configurar.

Segundo o tutorial, dispomos de todas as ferramentas necessárias para compilar o NGINX. Agora, precisamos de descarregar a fonte NGINX a partir do seu site oficial. Executei o seguinte comando: wget <http://nginx.org/download/nginx-1.20.0.tar.gz> (esse é o comando do tutorial)

Mas, pesquisando a última versão estável o nome do arquivo compactado atualmente é: nginx-1.22.1.tar.gz.

Portanto, executarei o comando:

wget <http://nginx.org/download/nginx-1.22.1.tar.gz>

```
letsubuntu@letsubuntu:~$ wget http://nginx.org/download/nginx-1.22.1.tar.gz
```

```
letsubuntu@letsubuntu:~$ ls  
nginx-1.22.1.tar.gz  
letsubuntu@letsubuntu:~$
```

Para extrair, usa-se o comando (tar -zxvf nginx-1.22.1.tar.gz)

```

nginx-1.22.1/objs/
letsubuntu@letsubuntu:~$ ls
nginx-1.22.1  nginx-1.22.1.tar.gz
letsubuntu@letsubuntu:~$

```

Para ir para dentro do diretório extraído, utiliza-se o comando (cd nginx-1.22.1)

Em seguida, utilizar a flag de configuração para configurar o NGINX:

```

./configure --prefix=/var/www/html --sbin-path=/usr/sbin/nginx --
conf-path=/etc/nginx/nginx.conf --http-log-
path=/var/log/nginx/access.log --error-log-
path=/var/log/nginx/error.log --with-pcre --lock-
path=/var/lock/nginx.lock --pid-path=/var/run/nginx.pid --with-
http_ssl_module --with-http_image_filter_module=dynamic --modules-
path=/etc/nginx/modules --with-http_v2_module --with-
stream=dynamic --with-http_addition_module --with-http_mp4_module

```

No comando acima, configurou-se o caminho personalizado para o arquivo de configuração NGINX, acesso, e caminho de registo de erros com algum módulo do NGINX.

Passo 3 - Build NGINX e Módulos adicionais

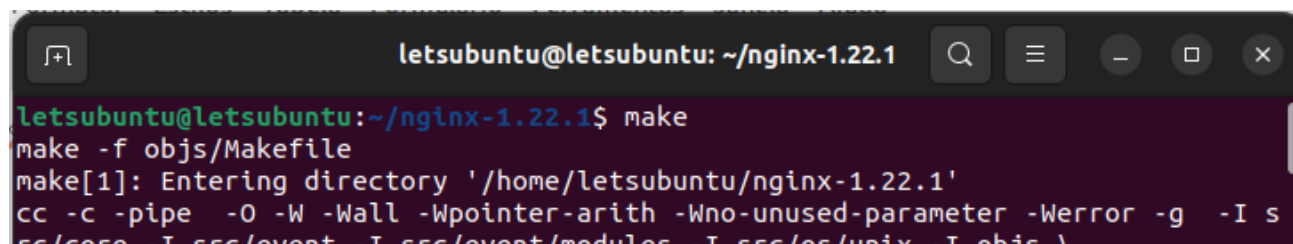
Há muitas opções de configuração disponíveis no NGINX, pode utilizá-lo de acordo com as suas necessidades. Para encontrar todas as opções de configuração disponíveis em NGINX, o tutorial recomenda acessar a documentação (<http://nginx.org/en/docs/configure.html>).

Módulos construídos por padrão

Muitos módulos vêm com o NGINX pré-instalado. Se não precisar de um módulo construído por padrão, pode desativá-lo, nomeando-o com a opção --without-<MODULE-NAME> no script de configuração.

Compilando código fonte do NGINX.

Após a configuração personalizada concluída, podemos agora compilar o código fonte NGINX



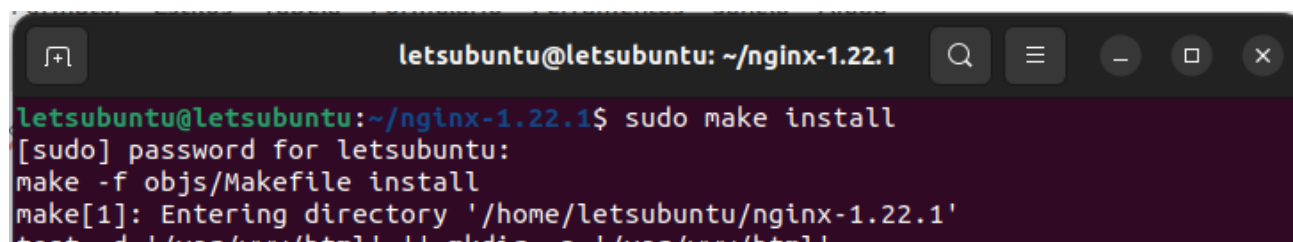
```

letsubuntu@letsubuntu: ~/nginx-1.22.1
letsubuntu@letsubuntu:~/nginx-1.22.1$ make
make -f objs/Makefile
make[1]: Entering directory '/home/letsubuntu/nginx-1.22.1'
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I s
cc/core -I src/event -I src/event/modules -I src/os/unix -I objs \

```

utilizando este comando: “make”.

Isto levará bastante tempo e, uma vez terminado, instalar o código fonte compilado, usando o comando (sudo make install).



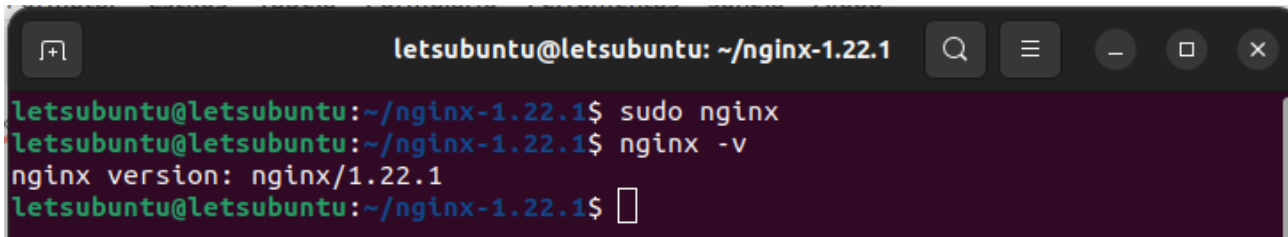
```

letsubuntu@letsubuntu: ~/nginx-1.22.1
letsubuntu@letsubuntu:~/nginx-1.22.1$ sudo make install
[sudo] password for letsubuntu:
make -f objs/Makefile install
make[1]: Entering directory '/home/letsubuntu/nginx-1.22.1'
test -d '/var/www/html' || mkdir -p '/var/www/html'

```

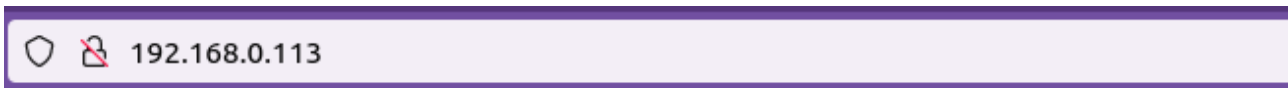
Iniciei o NGINX através do comando (sudo nginx)

Agora que o NGINX foi instalado com sucesso, é possível verificar através deste comando: “nginx -v”

A terminal window titled 'letsubuntu@letsubuntu: ~/nginx-1.22.1' with standard window controls. The terminal shows the following commands and output:

```
letsubuntu@letsubuntu:~/nginx-1.22.1$ sudo nginx
letsubuntu@letsubuntu:~/nginx-1.22.1$ nginx -v
nginx version: nginx/1.22.1
letsubuntu@letsubuntu:~/nginx-1.22.1$
```

Ou visitar o IP que mostrará a página do NGINX:
http://192.168.0.113



Welcome to nginx!

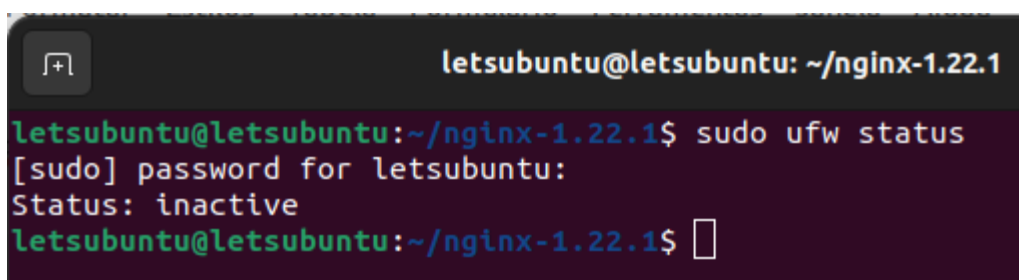
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Isso finaliza as questões Q2 e Q3.

Antes de ir para a questão 4, deixarei registradas algumas informações que são importantes. Sempre verificar o firewall, no caso da minha máquina virtual está inativo:

A terminal window titled 'letsubuntu@letsubuntu: ~/nginx-1.22.1' with standard window controls. The terminal shows the following commands and output:

```
letsubuntu@letsubuntu:~/nginx-1.22.1$ sudo ufw status
[sudo] password for letsubuntu:
Status: inactive
letsubuntu@letsubuntu:~/nginx-1.22.1$
```

Caso estivesse ativo, seria necessária a liberação das portas 443 (https) e 80 (http), através deste comando: sudo ufw allow 443 e sudo ufw allow 80.

Outro ponto importante é a automatização de processo, através da configuração do serviço de sistema para NGINX.

Podemos confirmar que o NGINX está funcionando, verificando o processo (ps aux | grep nginx).

```
letsubuntu@letsubuntu: ~/nginx-1.22.1
letsubuntu@letsubuntu:~/nginx-1.22.1$ ps aux | grep nginx
root      13497  0.0  0.0  8496  824 ?        Ss   22:34   0:00 nginx: master process nginx
nobody    13498  0.0  0.2  9188  3952 ?        S    22:34   0:00 nginx: worker process
letsubu+  13521  0.0  0.0  6300  724 pts/0    S+   23:12   0:00 grep --color=auto nginx
letsubuntu@letsubuntu:~/nginx-1.22.1$
```

Nota-se o processo master e worker acima.

Para mandar um sinal de parada, utiliza-se o comando (sudo nginx -s stop)

Ao tentar visitar o IP do servidor (<http://192.168.0.113>), pode-se confirmar que parou.



Não foi possível conectar

Ocorreu um erro durante uma conexão com 192.168.0.113.

- Este site pode estar temporariamente indisponível ou sobrecarregado. Tente novamente daqui a pouco.
- Se você não conseguir carregar nenhuma página, verifique a conexão de rede do computador.
- Se a rede ou o computador estiver protegido por um firewall ou proxy, verifique se o Firefox está autorizado a acessar a web.

Tentar novamente

A ideia é adicionar o systemd service.

O tutorial indica criar o arquivo no editor nano (sudo nano /lib/systemd/system/nginx.service)

A imagem a seguir ilustra o arquivo com o conteúdo indicado pelo tutorial

```
GNU nano 4.8 /lib/systemd/system/nginx.service
[Unit]
Description=The NGINX HTTP and reverse proxy server
After=syslog.target network-online.target remote-fs.target nss-lookup.target
Wants=network-online.target

[Service]
Type=forking
PIDFile=/var/run/nginx.pid
ExecStartPre=/usr/sbin/nginx -t
ExecStart=/usr/sbin/nginx
ExecReload=/usr/sbin/nginx -s reload
ExecStop=/bin/kill -s QUIT $MAINPID
PrivateTmp=true

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

Pode-se trocar a localização do PIDfile de acordo com o path do nosso arquivo de configuração customizado.

Para iniciar o systemd, utiliza-se o comando: (sudo systemctl restart nginx)

```
letsubuntu@letsubuntu: ~/nginx-1.22.1
letsubuntu@letsubuntu:~/nginx-1.22.1$ sudo systemctl restart nginx
letsubuntu@letsubuntu:~/nginx-1.22.1$
```

Agora pode gerir o seu NGINX utilizando Systemd. Pode também verificar o estado do NGINX, quer esteja ou não em execução, utilizando este comando (sudo systemctl status nginx)

```
letsubuntu@letsubuntu: ~/nginx-1.22.1
letsubuntu@letsubuntu:~/nginx-1.22.1$ sudo systemctl status nginx
● nginx.service - The NGINX HTTP and reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; disabled; vendor preset: enabled)
   Active: active (running) since Mon 2023-01-23 23:25:02 UTC; 1min 33s ago
     Process: 13566 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
     Process: 13577 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
    Main PID: 13578 (nginx)
```

Agora, como mencionamos, o outro recurso muito útil de um serviço systemd é permitir que o NGINX inicie automaticamente quando o sistema inicializar no momento, quando esta máquina for desligada ou reinicializada, o NGINX não estará mais em execução.

Obviamente não é bom para um servidor web em particular.

Portanto, para ativar a inicialização na inicialização, execute este comando.

(systemctl enable nginx)

```
letsubuntu@letsubuntu: ~/nginx-1.22.1
letsubuntu@letsubuntu:~/nginx-1.22.1$ sudo systemctl enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /lib/systemd/system/nginx.service.
letsubuntu@letsubuntu:~/nginx-1.22.1$
```

Q4) Esta questão pede que seja apresentado o passo a passo para a geração das chaves pública para dar suporte à conexão https.

Para criar a chave, segui o tutorial disponível neste link:

<https://techexpert.tips/pt-br/nginx-pt-br/habilitar-https-no-nginx/>

Por padrão o openssl vem instalado no nginx, caso não o fosse, seria necessário aplicar os comandos: (sudo apt-get update) e (sudo apt-get install nginx openssl).

Como já vem instalado, vamos direto para a criação de um diretório que conterá os certificados:

(sudo mkdir /etc/nginx/certificate)

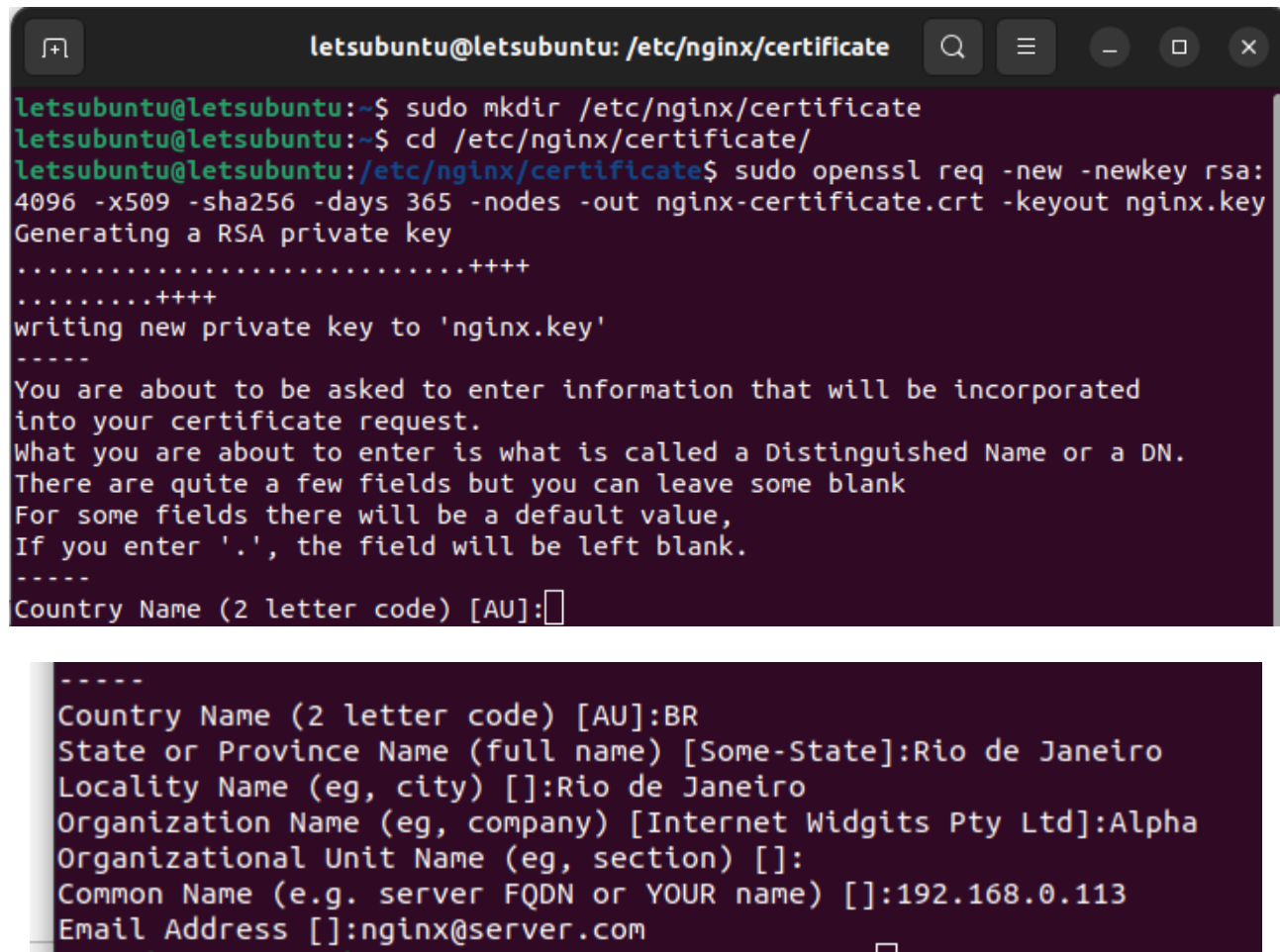
Acessando o diretório recém criado:

(cd /etc/nginx/certificate)

Criando a chave autoassinada através do openssl:

(sudo openssl req -new -newkey rsa:4096 -x509 -sha256 -days 365 -nodes -out nginx-certificate.crt -keyout nginx.key)

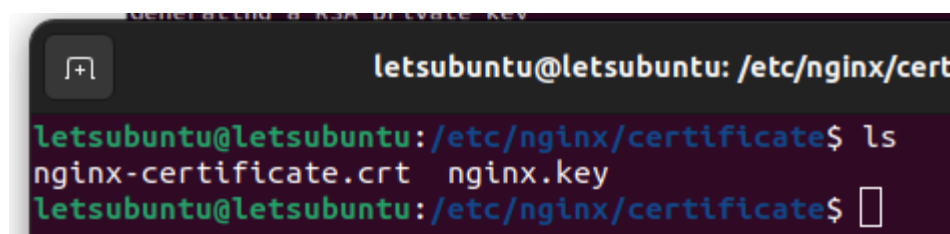
A imagem abaixo ilustra os três últimos comandos. Vão pedir algumas informações. É possível deixar várias em branco, no entanto, o campo Common Name deve ser preenchido com o IP do servidor ou o nome de domínio. Nesse caso, será inserido o IP da máquina remota (192.168.0.113)



```
letsubuntu@letsubuntu: /etc/nginx/certificate
letsubuntu@letsubuntu:~$ sudo mkdir /etc/nginx/certificate
letsubuntu@letsubuntu:~$ cd /etc/nginx/certificate/
letsubuntu@letsubuntu:/etc/nginx/certificate$ sudo openssl req -new -newkey rsa:4096 -x509 -sha256 -days 365 -nodes -out nginx-certificate.crt -keyout nginx.key
Generating a RSA private key
.....+++++
.....+++++
writing new private key to 'nginx.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
```

```
-----
Country Name (2 letter code) [AU]:BR
State or Province Name (full name) [Some-State]:Rio de Janeiro
Locality Name (eg, city) []:Rio de Janeiro
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Alpha
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:192.168.0.113
Email Address []:nginx@server.com
```

Pronto, as chaves foram criadas!



```
letsubuntu@letsubuntu: /etc/nginx/certificate
letsubuntu@letsubuntu:/etc/nginx/certificate$ ls
nginx-certificate.crt  nginx.key
letsubuntu@letsubuntu:/etc/nginx/certificate$
```

Q5) Ainda no mesmo tutorial, disponível em:

<https://techeexpert.tips/pt-br/nginx-pt-br/habilitar-https-no-nginx/>

é mostrado como é o arquivo de configuração antes e depois das modificações.

A imagem abaixo é o arquivo antes de alterar:

sudo vi /etc/nginx/sites-available/default

```
letsubuntu@letsubuntu: /etc/nginx/certificate
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    root /var/www/html;
    index index.html index.htm index.nginx-debian.html;
    server_name _;
    location / {
        try_files $uri $uri/ =404;
    }
}
```

Escolheu-se na alteração proceder da seguinte forma, definir a conexão de forma segura (https) e redirecionar eventuais acessos via http, de modo que vão para https.
Após a alteração:

```
GNU nano 4.8 /etc/nginx/sites-available/default Modified
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    server_name _;
    return 301 https://$host$request_uri;
}
server {
    listen 443 ssl default_server;
    listen [::]:443 ssl default_server;
    ssl_certificate /etc/nginx/certificate/nginx-certificate.crt;
    ssl_certificate_key /etc/nginx/certificate/nginx.key;
    root /var/www/html;
    index index.html index.htm index.nginx-debian.html;
    server_name _;
    location / {
        try_files $uri $uri/ =404;
    }
}
```

O código 301 é para redirecionamento. Algumas variáveis de ambiente estão presentes (\$host, \$uri, \$request_uri)

Nota-se que o root está em /var/www/html.

Falta reiniciar o serviço:
sudo service nginx restart

Com o serviço reiniciado, basta tentar acessar o endereço usando a versão https:

<https://192.168.0.113/>

Não deu certo! Após muito tempo tentando saber o que faltava ser mexido, Gabriela (Hopper) descobriu que deveríamos mexer no arquivo:

/etc/nginx/nginx.conf

Portanto, através do comando “sudo nano /etc/nginx/nginx.conf”, observou-se nesse arquivo algumas linhas que precisam ser removidos seus respectivos comentários (#) e algumas modificações.

```
server {
    listen      80;
    server_name localhost;

    #charset koi8-r;

    #access_log logs/host.access.log main;

    location / {
        root    /var/www/html;
        index   index.html index.htm;
    }

    #error_page  404              /404.html;
```

Na imagem acima, a primeira modificação foi mudar o root, O meu arquivo index.html personalizado tinha o seguinte caminho: /var/www/html/index.html.

Portanto, o novo root foi: “root /var/www/html;”

A próxima imagem apresenta outro local de mudança

```
# HTTPS server
#
server {
    listen      443 ssl;
    server_name localhost;

    ssl_certificate      /etc/nginx/certificate/nginx-certificate.crt;
    ssl_certificate_key  /etc/nginx/certificate/nginx.key;

    #    ssl_session_cache    shared:SSL:1m;
    #    ssl_session_timeout  5m;

    #    ssl_ciphers  HIGH:!aNULL:!MD5;
    #    ssl_prefer_server_ciphers  on;

    location / {
        root    /var/www/html;
        index   index.html index.htm;
    }
}
}
```

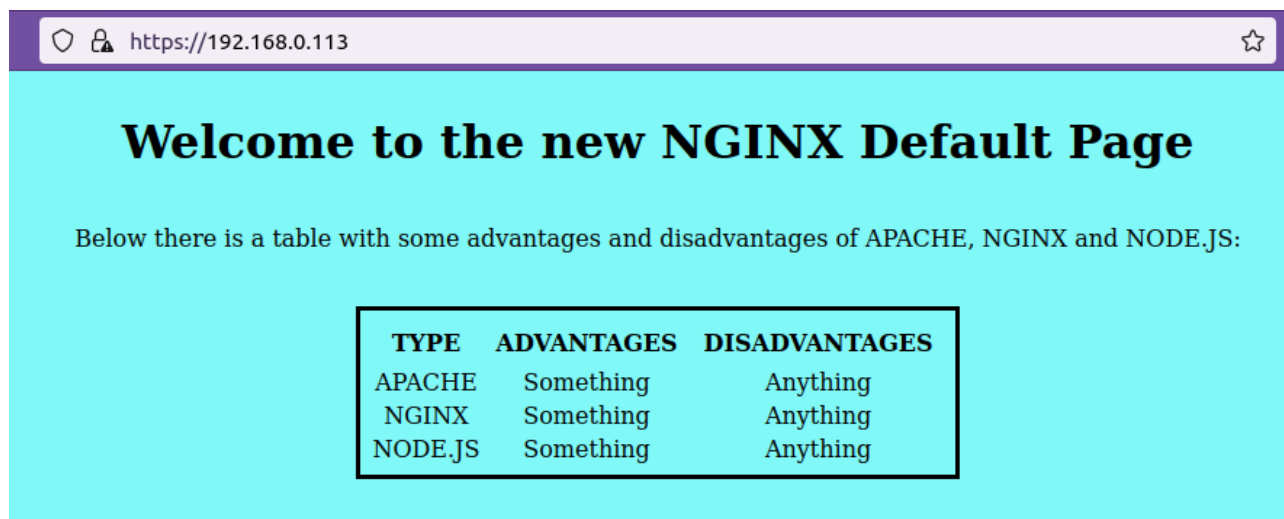
Algumas linhas precisaram ser descomentadas.

Além disso, nas linhas `ssl_certificate` e `ssl_certificate_key`, foi necessário mudar para apontar para onde estão as chaves que foram criadas. A imagem acima já está apontando para o local correto. Por fim, a última mudança necessária foi no root (a mesma ideia mostrada na porta 80).
“root /var/www/html;”

Salvei as alterações no arquivo e reiniciei o servidor:

“sudo service nginx restart”

Ao tentar acessar pelo protocolo <https://192.168.0.113> O resultado foi primeiro uma página avisando sobre potencial risco, e indo em opções avançadas clico em prosseguir. Finalmente, a página é mostrada a seguir:



Q6) A ideia aqui é semelhante ao que foi feito na aula passada (aula 04). Criar um usuário, deixar ele com acesso apenas a sua própria pasta pessoal.

A primeira etapa é a instalação do vsftpd (very secure FTP daemon).

A sequência de comandos apresentadas a seguir são provenientes do link:

https://docs.google.com/document/d/1_nuhFHHIOLqOcIHZkvGmS2pbCK-0uID7OObaogSAnAQ/edit#

Fornecido pelo professor.

1 - Atualização de dependências e pacotes

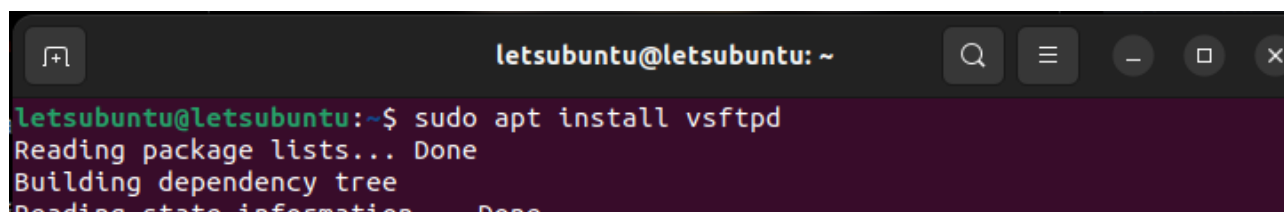
`sudo apt update && sudo apt upgrade`

2 - Instalar vsftpd

`sudo apt install vsftpd`

3 - Faça um backup do arquivo de configuração

`sudo cp /etc/vsftpd.conf /etc/vsftpd.conf.orig`



4-Verificando o firewall: `sudo ufw status`

```
letsubuntu@letsubuntu: ~  
letsubuntu@letsubuntu:~$ sudo ufw status  
Status: inactive  
letsubuntu@letsubuntu:~$
```

O firewall não está ativo, portanto, não é necessário habilitar as portas. Caso fosse necessário, coloca-se “sudo ufw allow 20” e “sudo ufw allow 21”. Essas são as portas relacionadas com o FTP. Deixando registrado, seria:

```
letsubuntu@letsubuntu:~$ sudo ufw allow 20  
Rules updated  
Rules updated (v6)  
letsubuntu@letsubuntu:~$ sudo ufw allow 21  
Rules updated  
Rules updated (v6)  
letsubuntu@letsubuntu:~$ sudo ufw allow 990/tcp  
Rules updated  
Rules updated (v6)  
letsubuntu@letsubuntu:~$ sudo ufw allow 40000:50000/tcp  
Rules updated  
Rules updated (v6)
```

A próxima etapa é configurar o usuário e os diretórios.

O FTP geralmente é mais seguro quando os usuários estão restritos a um diretório específico. Isso é realizado com vsftpd [chroot](#) jails. Quando está habilitado para usuários locais, eles são restritos ao diretório base por padrão. Como protege o diretório de uma maneira específica, ele não deve ser gravável pelo usuário. Isso é bom para um novo usuário que só deve se conectar via FTP, mas um usuário existente pode precisar gravar em sua pasta base se também tiver acesso ao shell.

0 - Criar um grupo:

```
sudo addgroup "teste-ftp"
```

1 - Adicione um usuário:

```
sudo adduser enzo
```

2 - Criar um diretório raiz para o envio de arquivos.

```
sudo mkdir /home/enzo/ftp
```

3 - Definir a propriedade

```
sudo chown -R enzo:enzo /home/enzo/ftp
```

4 - Atribuir permissões

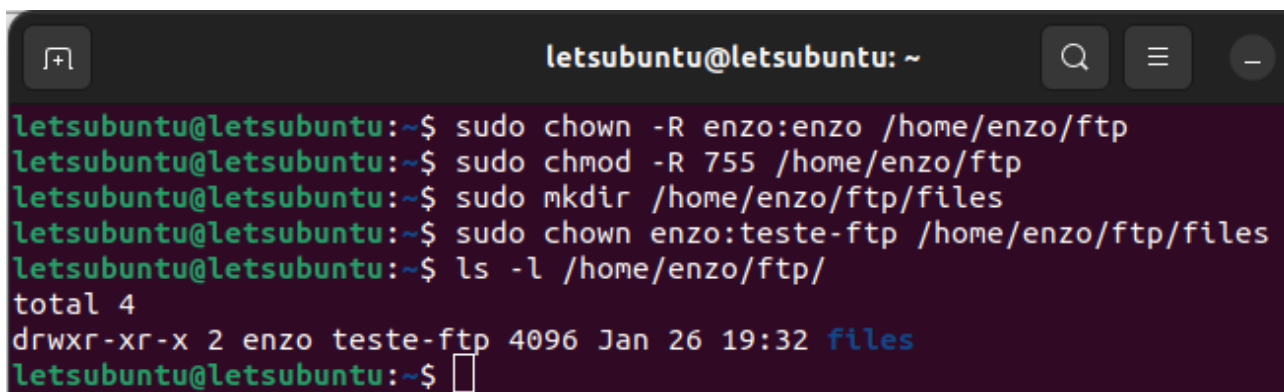
```
sudo chmod -R 755 /home/enzo/ftp
```

5 - Criar diretório específico para os arquivos

```
sudo mkdir /home/enzo/ftp/files
```

6 - Atribuir propriedades de usuário

```
sudo chown enzo:teste-ftp /home/enzo/ftp/files
```



```
letsubuntu@letsubuntu: ~  
letsubuntu@letsubuntu:~$ sudo chown -R enzo:enzo /home/enzo/ftp  
letsubuntu@letsubuntu:~$ sudo chmod -R 755 /home/enzo/ftp  
letsubuntu@letsubuntu:~$ sudo mkdir /home/enzo/ftp/files  
letsubuntu@letsubuntu:~$ sudo chown enzo:teste-ftp /home/enzo/ftp/files  
letsubuntu@letsubuntu:~$ ls -l /home/enzo/ftp/  
total 4  
drwxr-xr-x 2 enzo teste-ftp 4096 Jan 26 19:32 files  
letsubuntu@letsubuntu:~$
```

A próxima etapa é configurar o arquivo de configuração (sudo nano /etc/vsftpd.conf). Lembrando que as configurações originais estão no arquivo (sudo nano /etc/vsftpd.conf.orig), que serve como um backup caso seja necessário refazer.

Configuração do Acesso FTP

1 - Abrir o arquivo de configuração vsftpd.conf e apagar todas as informações.

```
sudo nano /etc/vsftpd.conf
```

2 - Substituir todas as informações pelos dados abaixo:

```
listen=YES
```

```
listen_ipv6=NO
```

```
anonymous_enable=NO
```

```
local_enable=YES
```

```
write_enable=YES
```

```
chroot_local_user=YES
```

```
allow_writeable_chroot=YES
```

```
secure_chroot_dir=/var/run/vsftpd/empty
```

```
user_sub_token=$USER
```

```
local_root=/home/$USER/ftp
```

```
pam_service_name=ftt
```

```
pasv_enable=YES
```

```
pasv_min_port=40000
```

```
pasv_max_port=50000
```

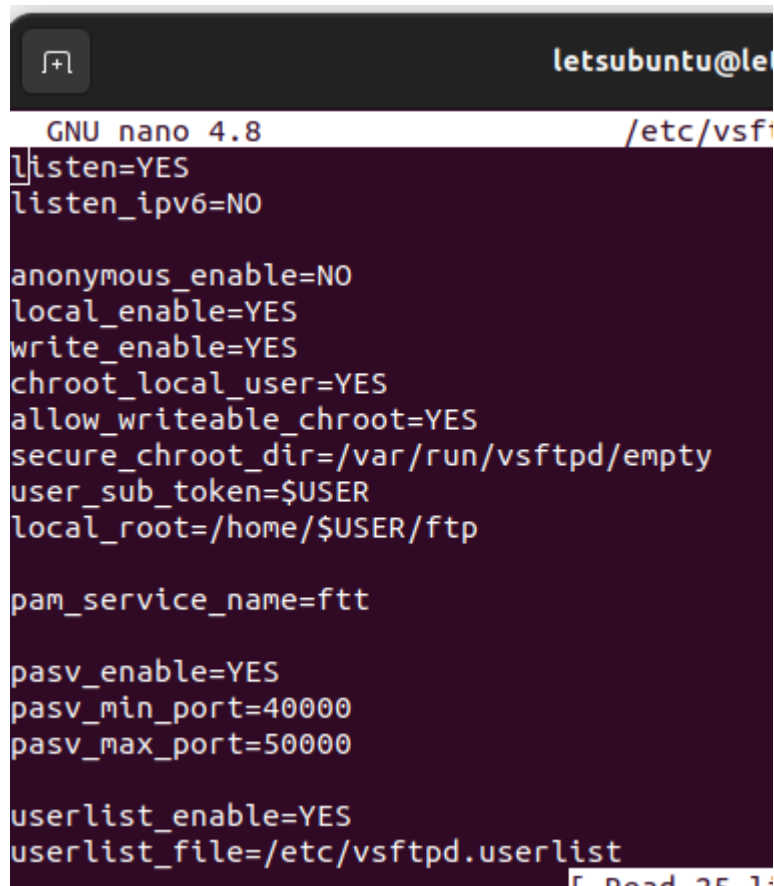
```
userlist_enable=YES
```

```
userlist_file=/etc/vsftpd.userlist
```

```
userlist_deny=NO
```

```
chroot_local_user=YES
chroot_list_enable=NO
chroot_list_file=/etc/vsftpd.chroot_list
```

A imagem a seguir mostra parte do arquivo após ter sido alterado.



```
GNU nano 4.8 /etc/vsftpd.conf
listen=YES
listen_ipv6=NO

anonymous_enable=NO
local_enable=YES
write_enable=YES
chroot_local_user=YES
allow_writeable_chroot=YES
secure_chroot_dir=/var/run/vsftpd/empty
user_sub_token=$USER
local_root=/home/$USER/ftp

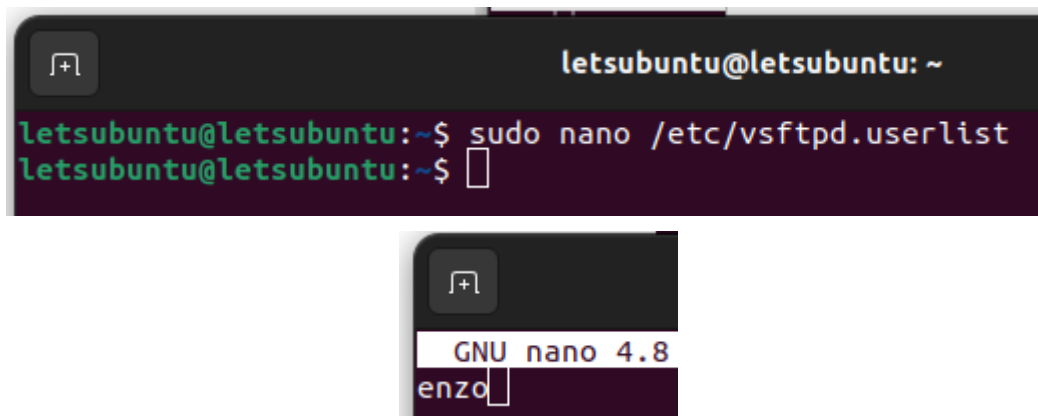
pam_service_name=ftpd

pasv_enable=YES
pasv_min_port=40000
pasv_max_port=50000

userlist_enable=YES
userlist_file=/etc/vsftpd.userlist
```

3 - Salvar o arquivo.

4 - Incluir usuário no arquivo /etc/vsftpd.userlist



```
letsubuntu@letsubuntu: ~
letsubuntu@letsubuntu:~$ sudo nano /etc/vsftpd.userlist
letsubuntu@letsubuntu:~$ 
GNU nano 4.8
enzo
```

5 - Reiniciar o servidor vsftpd e verificar o status

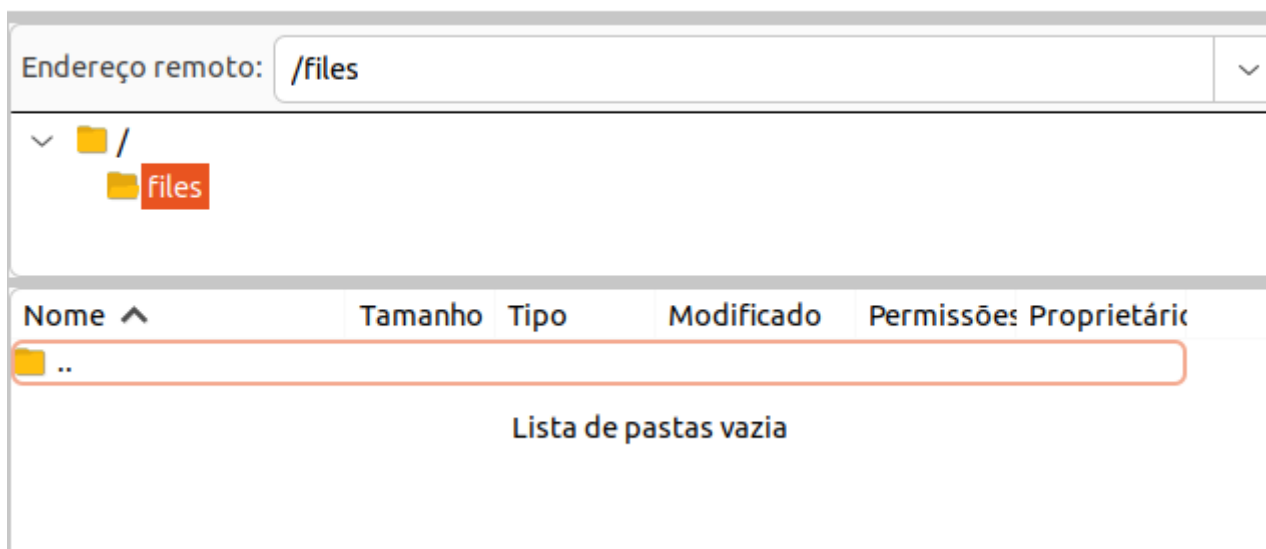
```
sudo systemctl restart vsftpd
```

```
sudo systemctl status vsftpd
```

```
letsubuntu@letsubuntu: ~  
letsubuntu@letsubuntu:~$ sudo systemctl restart vsftpd  
letsubuntu@letsubuntu:~$ sudo systemctl status vsftpd  
● vsftpd.service - vsftpd FTP server  
   Loaded: loaded (/lib/systemd/system/vsftpd.service; e  
   Active: active (running) since Thu 2023-01-26 19:41:0  
   Process: 2004 ExecStartPost= /bin/echo -n /usr/lib/vsft
```

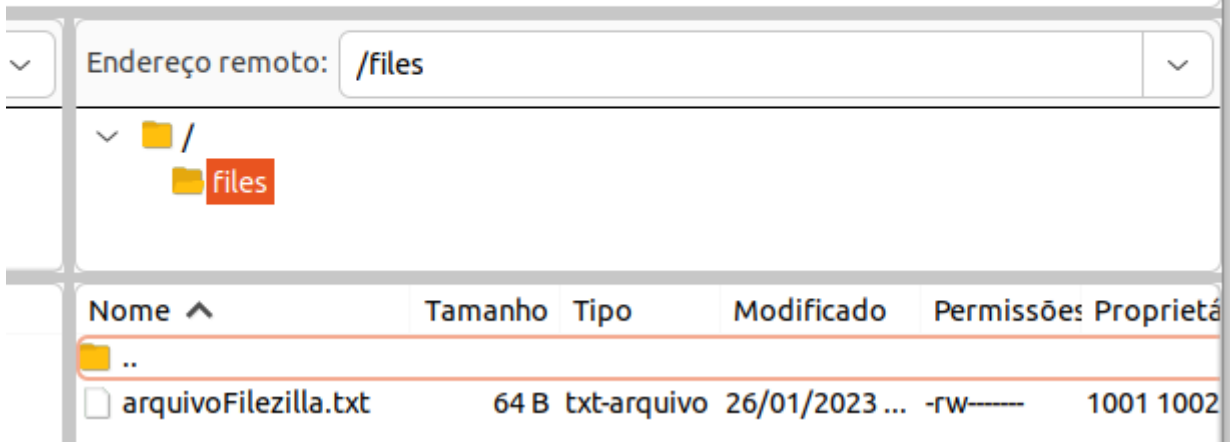
Pronto, está tudo certo para fazer a conexão ftp. Nesse caso mostrarei pelo filezilla
Lembrando que o nome do usuário é “enzo” e o IP da máquina remota que contém o NGINX é 192.168.0.113.

A conexão foi um sucesso e a pasta está vazia:



Adicionarei um arquivo que já usei na aula anterior (arquivoFilezilla.txt), cujo caminho local é /home/letonio/arquivoFilezilla.txt.

segundo



Agora, abrirei outro terminal, e utilizarei o comando “ftp enzo@192.168.0.113” para fazer a conexão via terminal.

```
letonio@letonio-Inspiron-15-3567: ~  
letonio@letonio-Inspiron-15-3567:~$ ftp enzo@192.168.0.113  
Connected to 192.168.0.113.  
220 (vsFTPd 3.0.3)  
331 Please specify the password.  
Password:  
230 Login successful.  
Remote system type is UNIX.  
Using binary mode to transfer files.  
ftp>
```

Para mostrar que está o usuário enzo está com chroot jail, note que ao usar o comando pwd não conseguimos ver nada. Ele terá apenas acesso ao seu diretório pessoal que possui apenas a pasta files.

```
letonio@letonio-Inspiron-15-3567:~$ ftp enzo@192.168.0.113  
Connected to 192.168.0.113.  
220 (vsFTPd 3.0.3)  
331 Please specify the password.  
Password:  
230 Login successful.  
Remote system type is UNIX.  
Using binary mode to transfer files.  
ftp> pwd  
Remote directory: /
```

Entrando na pasta files, nota-se que o arquivo que foi transferido via filezilla encontra-se lá.

```
letonio@letonio-Inspiron-15-3567: ~  
ftp> cd files  
250 Directory successfully changed.  
ftp> ls  
229 Entering Extended Passive Mode (|||48176|)  
150 Here comes the directory listing.  
-rw----- 1 1001 1002 64 Jan 26 19:47 arquivoFilezilla.txt  
226 Directory send OK.  
ftp> █
```

Para finalizar, utilizaremos o comando put para transferir o arquivo viaterminal.txt da máquina local para a máquina remota. Isso será feito pelo comando:

“put /home/letonio/viaterminal.txt /filex/viaterminal.txt”

```
ftp> put /home/letonio/viaterminal.txt /files/viaterminal.txt  
local: /home/letonio/viaterminal.txt remote: /files/viaterminal.txt  
229 Entering Extended Passive Mode (|||47457|)  
150 Ok to send data.  
100% |*****| 34 1.13 KiB/s 00:00 ETA  
226 Transfer complete.  
34 bytes sent in 00:00 (1.04 KiB/s)  
ftp> ls  
229 Entering Extended Passive Mode (|||48756|)  
150 Here comes the directory listing.  
-rw----- 1 1001 1002 64 Jan 26 19:47 arquivoFilezilla.txt  
-rw----- 1 1001 1002 34 Jan 26 19:55 viaterminal.txt  
226 Directory send OK.  
ftp> █
```

Pronto, nota-se que o arquivo está lá, caso alguém deseje transferir da máquina remota para a local, utiliza-se o comando get:

```
225 No transfer to ABOR.  
ftp> get /files/viaterminal.txt /home/letonio/Documentos/viaterminal.txt  
local: /home/letonio/Documentos/viaterminal.txt remote: /files/viaterminal.txt  
229 Entering Extended Passive Mode (|||45480|)  
150 Opening BINARY mode data connection for /files/viaterminal.txt (34 bytes).  
100% |*****| 34 851.36 KiB/s 00:00 ETA  
226 Transfer complete.  
34 bytes received in 00:00 (119.86 KiB/s)  
ftp> █
```

Nota-se que o arquivo da máquina remota (servidor) foi transferido para a máquina local:

```
letonio@letonio-Inspiron-15-3567: ~  
letonio@letonio-Inspiron-15-3567:~$ ls ~/Documentos/  
0_passwords  ExerciciosListaVitor  Livros  script.js  
alphaedtech  index.html             meutexto.txt  teste.png  
Design       lembrar-de-ler         myProjects   untitled_0.odt  
English      lideranca.txt          relembrandoConteudos  viaterminal.txt  
letonio@letonio-Inspiron-15-3567:~$ █
```