

Parcial Practico

Adjuntos [2023-03 Primer Parcial ServiciosTelematicos.pdf](#)

Tabla de contenido:

1. Configuración inicial de Apache:
2. Configurar Apache para `/archivos_privados`:
3. Módulos PAM para Apache:
4. Crear la lista de usuarios denegados:
5. Configurar restricciones de acceso en Apache:
Túnel hacia servidor web usando vagrant + ngrok
 1. Port Forwarding en Vagrant
 2. Instalar ngrok en el servidor Vagrant
 3. Configurar ngrok
 4. Agregar ngrok al PATH del sistema
 5. Validación
 6. Vagrant Share y ngrok en maquina host windows

1. Configuración inicial de Apache:

Del anterior taller ya tenemos nuestra configuración inicial de apache, por lo que solo es necesario iniciar el servicio apache:

```
systemctl start httpd
```

2. Configurar Apache para `/archivos_privados`:

2.1. Primero, creamos el directorio `/archivos_privados` en nuestro servidor:

```
sudo mkdir /var/www/html/archivos_privados
```

2.2. Ahora, vamos a editar el archivo de configuración de Apache para configurar la autenticación en el directorio

```
/archivos_privados. Usaremos el editor nano
```

Ahora, crearemos el archivo `index.html` dentro de `/archivos_privados`:

```
nano /var/www/html/archivos_privados/index.html
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Autenticación PAM - Acceso Seguro</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-rbsA2VBKQhggwzx"
</head>
<body>

<div class="container mt-5">
  <div class="row justify-content-center">
    <div class="col-md-6">
      <div class="card">
        <div class="card-header bg-primary text-white text-center">
          <h2><svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" fill="currentColor" class="bi bi-shield-lock" viewBo
          <path d="M5.338 1.59a61.44 61.44 0 0 0-2.837 8.56 48.1 48.1 0 0 0-.328 3.9c-.554 4.157 7.19 2.253 9.188a10.725 10.725 0 0 0 2.287 2.233c.
          <path d="M9.5 6.5a1.5 1.5 0 0 1-1 1.415l.385 1.99a.5.5 0 0 1-.491.595h-.788a.5.5 0 0 1-.49-.595l.384-1.99a1.5 1.5 0 1 1 2-1.415z"/>
        </svg> Autenticación PAM</h2>
        <small>Acceso Seguro</small>
      </div>
      <div class="card-body">
        <p class="lead">Bienvenido al directorio seguro.</p>
        <p>Este directorio está protegido por el módulo de autenticación PAM. Si estás viendo esta página, significa que has pa
      </div>
    </div>
  </div>
</div>
</div>
```

```
<!-- Latest Bootstrap bundle.js -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-kenU1KFdBIe4zVF0s0G1M5b4hcxpYD"
</body>
</html>
```

2.3. Añade la siguiente configuración al final del archivo:

```
sudo nano /etc/httpd/conf/httpd.conf
```

```
<Directory "/var/www/html/archivos_privados">
    AuthType Basic
    AuthName "Acceso restringido"
    AuthBasicProvider PAM
    AuthPAMService apache
    Require valid-user
</Directory>
```

Esto configurará la autenticación básica con PAM para el directorio `/archivos_privados`. La línea `AuthPAMService apache` le dice a Apache que utilice el servicio PAM llamado 'apache' (puede tomar cualquier otro nombre), que configuraremos en el próximo paso.

2.4. Una vez realizado esto y *instalado el módulo PAM*, reinicia el servicio Apache para que los cambios surtan efecto:

```
sudo systemctl restart httpd
```

Este paso establece la base para la autenticación de PAM. Ahora, los usuarios que intenten acceder al directorio `/archivos_privados` se encontrarán con un desafío de autenticación.

3. Módulos PAM para Apache:

Para hacer que Apache utilice PAM, es necesario instalar el módulo PAM para Apache y configurarlo.

Instrucciones:

3.1. Instala el módulo PAM para Apache. Dependiendo de tu sistema, el paquete podría llamarse `mod_authnz_pam` o `mod_auth_pam`. Dado que estamos usando CentOS 9, intentemos con el siguiente comando:

```
sudo dnf install -y mod_authnz_pam
```

```
root@servidor:/etc/yum.repos.d
Installed size: 31 k
Downloading Packages:
mod_authnz_pam-1.2.2-3.el9.x86_64.rpm                3.6 kB/s | 21 kB    00:05
-----
Total                                                1.8 kB/s | 21 kB    00:11
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :                                1/1
  Installing     : mod_authnz_pam-1.2.2-3.el9.x86_64 1/1
  Running scriptlet: mod_authnz_pam-1.2.2-3.el9.x86_64 1/1
  Verifying      : mod_authnz_pam-1.2.2-3.el9.x86_64 1/1

Installed:
  mod_authnz_pam-1.2.2-3.el9.x86_64

Complete!
[root@servidor yum.repos.d]#
```

3.2. Crea un archivo de configuración PAM para el servicio `apache` que especificamos anteriormente:

```
sudo nano /etc/pam.d/apache
```

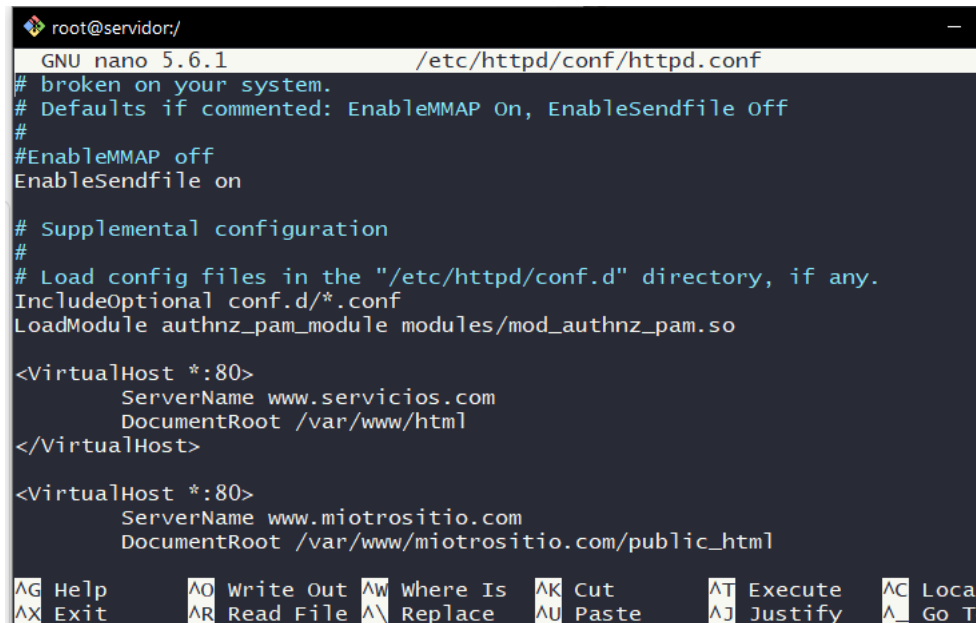
3.3. En el archivo, añade lo siguiente para permitir la autenticación de usuarios del sistema:

```
auth required pam_unix.so
account required pam_unix.so
```

Guarda el archivo y cierra el editor.

ahora necesitamos agregar explícitamente la carga del módulo en la configuración de Apache si no se hizo automáticamente durante la instalación. Añade la siguiente línea en tu archivo `/etc/httpd/conf/httpd.conf`:

```
LoadModule authnz_pam_module modules/mod_authnz_pam.so
```



```
root@servidor:/
GNU nano 5.6.1 /etc/httpd/conf/httpd.conf
# broken on your system.
# Defaults if commented: EnableMMAP On, EnableSendfile Off
#
#EnableMMAP off
EnableSendfile on

# Supplemental configuration
#
# Load config files in the "/etc/httpd/conf.d" directory, if any.
IncludeOptional conf.d/*.conf
LoadModule authnz_pam_module modules/mod_authnz_pam.so

<VirtualHost *:80>
    ServerName www.servicios.com
    DocumentRoot /var/www/html
</VirtualHost>

<VirtualHost *:80>
    ServerName www.miotrositio.com
    DocumentRoot /var/www/miotrositio.com/public_html
</VirtualHost>

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute  ^C Loca
^X Exit      ^R Read File ^N Replace   ^U Paste     ^J Justify  ^_ Go T
```

vamos a autenticar el acceso al servicio Apache usando las cuentas Linux.

Habilitar el servicio Apache para leer el archivo SHADOW.

- Crear el grupo shadow si no existe

```
sudo groupadd shadow
```

- Agregar el usuario Apache al grupo shadow

```
sudo usermod -a -G shadow apache
```

- Cambiar el propietario y los permisos del archivo `/etc/shadow` para que el grupo shadow pueda leerlo

```
sudo chown root:shadow /etc/shadow
sudo chmod g+r /etc/shadow
```

- Reiniciar el servicio Apache

```
sudo systemctl restart httpd
```

PAM authentication Test

- Crear un usuario de Linux para pruebas de autenticación, si es necesario

```
sudo adduser admin
sudo passwd admin
```

En nuestro ejemplo, creamos una cuenta Linux llamada ADMIN.

La contraseña configurada fue 123qwe.

3.4. Reinicia el servicio Apache para que los cambios surtan efecto:

```
sudo systemctl restart httpd
```

Este paso asegura que Apache utilice PAM para la autenticación, y el servicio `httpd` especificado usará `pam_unix.so` para autenticar contra los usuarios del sistema.



Si SELinux está activado, cambie la política de la siguiente manera: `setsebool -P httpd_mod_auth_pam on`

Probamos con `pamtester`:

```
root@servidor:/
[root@servidor /]# pamtester httpd user1 authenticate
Password:
pamtester: successfully authenticated
[root@servidor /]#

root@servidor:/
[root@servidor /]# pamtester httpd vagrant authenticate
Password:
pamtester: successfully authenticated
[root@servidor /]#
```

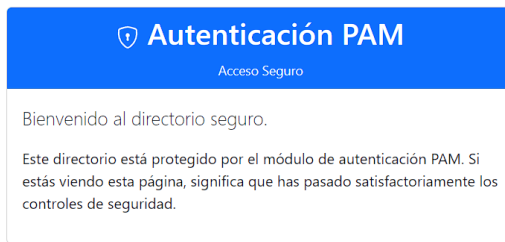
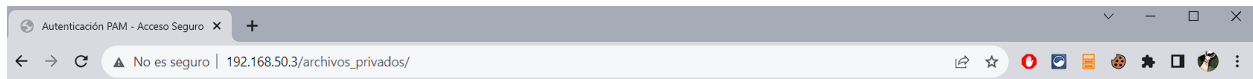
Probamos con `curl`:

`curl -u admin:123qwe http://192.168.50.3/archivos_privados/`

```
root@cliente:/
[root@cliente /]# curl -u admin:123qwe http://192.168.50.3/archivos_privados/
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Autenticación PAM - Acceso Seguro</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zw1RWuH61DGLwZJEdK2Kadq2F9CUG65" crossorigin="anonymous">
</head>
<body>

<div class="container mt-5">
  <div class="row justify-content-center">
    <div class="col-md-6">
      <div class="card">
        <div class="card-header bg-primary text-white text-center">
          <h2><svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" fill="currentColor" class="bi bi-shield-lock" viewBox="0 0 16 16">
            <path d="M5.338 1.59a61.44 61.44 0 0 0-2.837 8.56 481.481 0 0 0-.328 39c-.554 4.157-.726 7.19 2.253 9.188a10.725 10.725 0 0 0 2.287 2.233c.346.244.652.42 893.53 3.12.057.218.095.293.118a.55.55 0 0 0 .101.025.615.615 0 0 0 .1-.025c.076-.023.174-.061.294-.118.24-.113.547-.29.893-.533a10.726 10.726 0 0 0 2.287-2.233c1.527-
```

en el browser del host:



4. Crear la lista de usuarios denegados:

Queremos que ciertos usuarios del sistema sean explícitamente denegados el acceso al directorio `/archivos_privados`. Para ello, almacenaremos los nombres de estos usuarios en un archivo separado.

4.1. Crear un archivo para contener la lista de usuarios denegados. Podemos poner este archivo en `/etc/httpd/` para mantenerlo junto con otras configuraciones de Apache:

```
sudo nano /etc/httpd/denegados.txt
```

4.2. En este archivo, añade los nombres de usuario que deseas denegar, uno por línea. Por ejemplo:

```
user1
user2
```

Guarda el archivo y sal del editor.

4.3. Modifica los permisos del archivo para asegurar que solo el usuario raíz pueda leerlo y escribir en él:

```
sudo chmod 600 /etc/httpd/denegados.txt
```

Este paso crea la lista de usuarios que serán denegados el acceso al directorio `/archivos_privados`.

5. Configurar restricciones de acceso en Apache:

En este paso, modificaremos la configuración de Apache para leer la lista de usuarios denegados y restringir el acceso al directorio `/archivos_privados` de acuerdo a ella.

Instrucciones:

1. Configurar el servicio PAM `apache`:

```
sudo nano /etc/pam.d/apache
```

agregamos la siguiente línea:

```
auth required pam_listfile.so item=user sense=deny file=/etc/httpd/denegados.txt onerr=succeed
```

y cambiamos los `required` de `auth` y `account` por `include`:

```
root@servidor:/# nano /etc/pam.d/apache
auth required pam_listfile.so item=user sense=deny file=/etc/httpd/denegados.txt onerr=succeed
auth include pam_unix.so
account include pam_unix.so
```

2. Establecer permisos para leer /etc/shadow

```
sudo chgrp apache /etc/shadow
sudo chmod 440 /etc/shadow
```

3. Recargar apache

```
sudo systemctl restart httpd
```

4. probamos desde nuestra maquina cliente:

lista de denegados:

```
root@servidor:/# nano /etc/httpd/denegados.txt
user1
user2
```

curl -u user1:Diminombre21 http://192.168.50.3/archivos_privados/

```
root@cliente:/# curl -u user1:Diminombre21 http://192.168.50.3/archivos_privados/
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>401 Unauthorized</title>
</head><body>
<h1>Unauthorized</h1>
<p>This server could not verify that you
are authorized to access the document
requested. Either you supplied the wrong
credentials (e.g., bad password), or your
browser doesn't understand how to supply
the credentials required.</p>
</body></html>
[root@cliente /]#
```

Explicación:

- **Apache Configuration:** Hemos configurado el archivo `httpd.conf` para aplicar autenticación básica en el directorio `/archivos_privados`. Se ha especificado que utilice PAM para la autenticación con `AuthPAMService apache`.
- **Módulos PAM:** Utilizamos el módulo `pam_unix.so` para permitir solo a los usuarios del sistema autenticarse.
- **Lista de usuarios denegados:** Se mantuvo un archivo `/etc/httpd/denegados.txt` que contiene una lista de usuarios que deberían ser denegados el acceso.

Túnel hacia servidor web usando vagrant + ngrok

1. Port Forwarding en Vagrant

Edita tu `Vagrantfile` para añadir una regla de port forwarding. Esto permitirá que el puerto 80 de tu máquina virtual se mapee al puerto 8080 de tu máquina host (o cualquier otro puerto que prefieras).

Añade la siguiente línea en tu **Vagrantfile**:

```
config.vm.network "forwarded_port", guest: 80, host: 8080
```

quedando así por ejemplo:

```
config.vm.define :servidor do |servidor|
  servidor.vm.box = "generic/centos9s"
  servidor.vm.network :private_network, ip: "192.168.50.3"
  servidor.vm.hostname = "servidor"
  servidor.vm.network "forwarded_port", guest: 80, host: 8080
end
```

Reinicia tu máquina Vagrant para aplicar los cambios:

```
vagrant reload
```

2. Instalar ngrok en el servidor Vagrant

Para instalar ngrok, primero debes acceder a tu máquina Vagrant con SSH:

```
vagrant ssh servidor
```

Una vez dentro de la máquina, puedes descargar e instalar ngrok con los siguientes comandos:

Activar snaps en CentOS e instalar ngrok

Los Snaps son aplicaciones empaquetadas con todas sus dependencias para ejecutarse en todas las distribuciones populares de Linux desde una única compilación. Se actualizan automáticamente.



Snap está disponible para CentOS 7.6+, y Red Hat Enterprise Linux 7.6+, desde el repositorio Extra Packages for Enterprise Linux (EPEL). El repositorio EPEL puede ser añadido a su sistema con el siguiente comando:

```
sudo yum install epel-release
```

```
sudo yum install snapd
```

```
sudo systemctl enable --now snapd.socket
```

```
sudo ln -s /var/lib/snapd/snap /snap
```

ahora accedemos a nuestro paquete **ngrok** instalado

```
cd /
```

está ubicado en la carpeta **snap** y adentro de la subcarpeta **bin**

3. Configurar ngrok

Primero, necesitamos exponer el puerto 80 del servidor web Apache al mundo exterior usando ngrok. Asegúrate de estar aún dentro de la máquina virtual Vagrant a la que accediste mediante SSH.

Ejecuta el siguiente comando para iniciar ngrok y abrir un túnel para el puerto 80:

```
cd snap/bin && ./ngrok http 80
```

veremos algo así

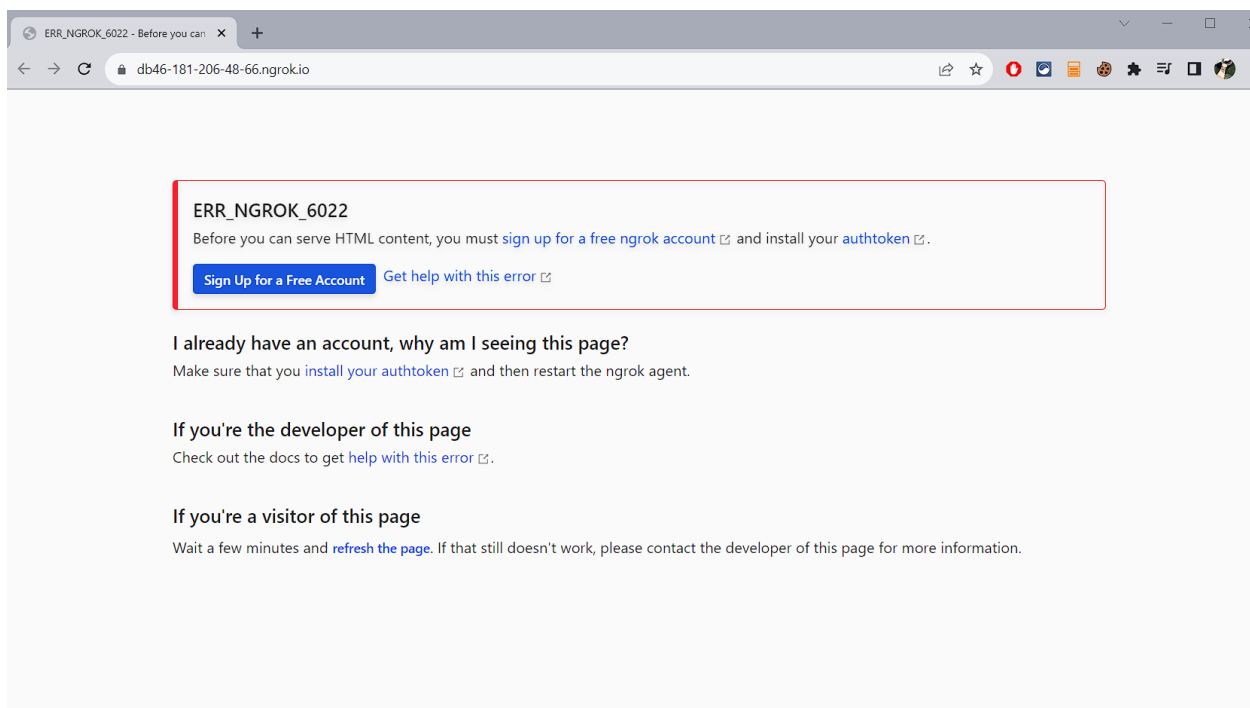
```
root@servidor:/snap/bin
ngrok (Ctrl+C to quit) ^

Take our ngrok in production survey! https://forms.gle/axiBFwzEA36DudFn6

Session Status      online
Session Expires     1 hour, 59 minutes
Update              update available (version 3.3.4, Ctrl-U to update)
Terms of Service     https://ngrok.com/tos
Version             3.3.1
Region              United States (us)
Latency              109ms
Web Interface        http://127.0.0.1:4040
Forwarding           https://db46-181-206-48-66.ngrok.io -> http://loca

Connections          ttl    opn    rt1    rt5    p50    p90
1                  1      0      0.01   0.00   5.70   5.70
```

y cuando accedemos al link:



tendremos que registrarnos y establecer nuestro Authtoken, con el siguiente comando y solo es sustituir el TOKEN por el Authtoken que nos dan en la pagina de ngrok

```
./ngrok config add-authtoken <TOKEN>
```

y tendremos que volver a lanzar la aplicacion

```
./ngrok http 80
```

4. Agregar ngrok al PATH del sistema

Para hacer que `ngrok` esté disponible globalmente en tu sistema, puedes agregarlo al PATH. Esto es especialmente útil si tienes planes de usar `ngrok` con regularidad.

Paso 1: Edita el archivo `.bashrc` o `.bash_profile` (dependiendo de tu shell y configuración).

Puedes hacerlo usando el editor `vi`, `nano`, o cualquier editor de tu preferencia:

```
vi ~/.bashrc
```

o


```
nano ~/.bashrc
```

Paso 2: Añadir la siguiente línea al final del archivo `~/.bashrc` o `~/.bash_profile`.

Paso 3: Aplicar los cambios.

Para aplicar los cambios en tu sesión actual, ejecuta:

```
source ~/.bashrc
```

O si usaste `~/.bash_profile`:

```
source ~/.bash_profile
```

Con estos pasos, `ngrok` debería estar ahora disponible en cualquier lugar del sistema, lo que facilita su uso.

5. Validación

Para validar que ngrok se ha añadido correctamente al PATH, puedes intentar ejecutar el siguiente comando en cualquier directorio:

```
ngrok --version
```

Si el comando se ejecuta con éxito y muestra la versión, entonces has configurado ngrok correctamente.

```
[root@servidor bin]# nano ~/.bashrc
[root@servidor bin]# source ~/.bashrc
[root@servidor bin]# ngrok --version
ngrok version 3.3.1
[root@servidor bin]# |
```

ahora tenemos ngrok funcionando en nuestro servidor podriamos realizar un script de inicio para ngrok y luego agregarlo al crontab, pero lo que vamos hacer es ahora utilizar el Vagrant Share.

6. Vagrant Share y ngrok en maquina host windows

Vagrant Share es un plugin de Vagrant que debe ser instalado. No se incluye con los paquetes del sistema Vagrant. Para instalar el plugin Vagrant Share, ejecute el siguiente comando:

```
vagrant plugin install vagrant-share
```

una vez instalado si ejecutamos vagrant share servidor --http 80, tendremos esto:

```
Santiago@ADMIN MINGW64 ~/Documents/prueba
$ vagrant plugin install vagrant-share
Installing the 'vagrant-share' plugin. This can take a few minutes...
Building native extensions. This could take a while...
Installed the plugin 'vagrant-share (2.0.0)!'

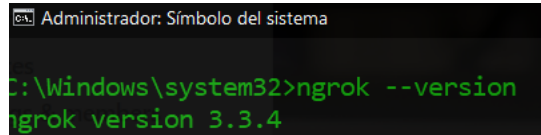
Santiago@ADMIN MINGW64 ~/Documents/prueba
$ vagrant share --http 80
The executable 'ngrok' Vagrant is trying to run was not found
in the PATH variable. The 'ngrok' executable is required to
run Vagrant share. If 'ngrok' is currently installed in a
non-standard location, append that location to the PATH
variable and run this command again. If 'ngrok' is not
currently installed, it can be downloaded from the ngrok
website:

https://ngrok.com/download
```

Por lo que tendremos que seguir los siguientes pasos

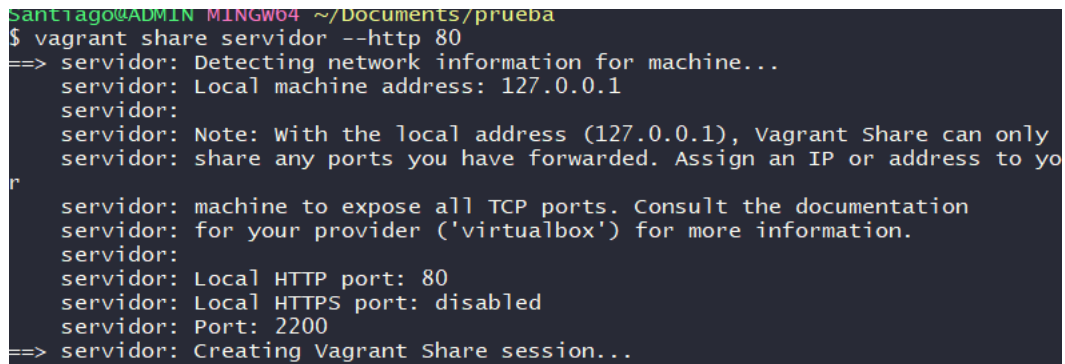
1. **Descargar ngrok en tu Máquina Host:** Ya que estás trabajando en Windows (según la salida de la terminal), ve al sitio web de ngrok y descarga el ejecutable de Windows. Una vez descargado, puedes ponerlo en una carpeta que esté en tu PATH o agregar la carpeta al PATH. aqui un [tutorial](#) por si aca.
 - **Agregar al PATH en Windows:**

1. Busca "Variables de entorno" en el menú de inicio y selecciónalo.
 2. En "Variables de usuario", busca la variable "PATH" y haz clic en "Editar".
 3. Agrega la carpeta donde se encuentra `ngrok.exe` y guárdalo.
2. **Verificar que ngrok Funciona en el Host:** Ejecuta `ngrok --version` en una nueva terminal para asegurarte de que se reconoce el comando.



```
Administrador: Símbolo del sistema
C:\Windows\system32>ngrok --version
ngrok version 3.3.4
```

3. **Reintentar vagrant share:** Ahora intenta nuevamente ejecutar `vagrant share servidor --http 80`.



```
santiago@ADMIN MINGW64 ~/Documents/prueba
$ vagrant share servidor --http 80
==> servidor: Detecting network information for machine...
servidor: Local machine address: 127.0.0.1
servidor:
servidor: Note: With the local address (127.0.0.1), Vagrant Share can only
servidor: share any ports you have forwarded. Assign an IP or address to yo
r
servidor: machine to expose all TCP ports. Consult the documentation
servidor: for your provider ('virtualbox') for more information.
servidor:
servidor: Local HTTP port: 80
servidor: Local HTTPS port: disabled
servidor: Port: 2200
==> servidor: Creating Vagrant Share session...
```

BONUS: [Link al github de implementacion blockchain](#)