

# Capturando datos con Wireshark

Por: Santiago Naranjo Herrera

## Índice

- Introducción
- Instalación De Las Dependencias Para Este Laboratorio
- Wireshark
  - Python
  - Very Secure FTP Daemon (vsftpd)
  - Un Cliente FTP
- Ngrok
- Primera Captura de Datos: Servidor HTTP
  - Montar El Servidor
  - Capturar Los Datos
- Segunda Captura: Servidor FTP
  - Abrir el túnel FTP
  - Conectarse a el servidor FTP y Enviar Archivos
  - Capturar los datos
- Conclusión

## Introducción

En el contexto de la seguridad informática y la administración de redes, la captura y análisis de datos en tránsito es una práctica fundamental para comprender el comportamiento de los protocolos de comunicación y detectar posibles vulnerabilidades. Este informe documenta un laboratorio en el cual se utiliza una máquina local con Arch Linux como puente para capturar y analizar datos transferidos entre dos servidores: uno configurado como servidor HTTP y otro como servidor FTP.

El propósito de este laboratorio es doble: primero, comprender cómo los datos se transmiten a través de estos protocolos en condiciones controladas; segundo, identificar y analizar las diferencias en la seguridad y en la estructura de los datos capturados entre HTTP y FTP. Utilizando herramientas como `nginx` para la configuración del servidor HTTP y un servidor FTP con las herramientas de `Very Secure FTP Daemon (vsftpd)` y `ngrok`.

En el servidor HTTP se capturaron datos de tipo HTTP que representan una petición no protegida que viaja a través de la red. Sin embargo, en el servidor FTP se capturaron datos de tipo FTP-DATA, debido a que la interceptación de datos ya no tiene énfasis en las consultas sino en los archivos que viajan a través de estas consultas.

## Instalación De Las Dependencias Para Este Laboratorio

Para este laboratorio se utilizaron diversas herramientas que sirvieron como un apoyo para la creación de los servidores HTTP y FTP. Cabe resaltar que este documento contiene los comandos de instalación para una computadora que hace uso del sistema Operativo Arch Linux.

### Wireshark

Para capturar los datos vamos a necesitar de una herramienta que sea capaz de llevar un registro del tráfico de red de la computadora. Es ahí cuando entra Wireshark, el cual cumple la función de analizar los paquetes y las distintas peticiones que se envían al momento de navegar por internet. Para instalar Wireshark en Arch Linux podemos usar el siguiente comando:

```
yay -S wireshark-git
```

Es importante mencionar que para acceder al registro de los datos de la tarjeta de red se debe iniciar el programa mediante el modo de administrador

### Python

Para crear un servidor HTTP simple que soporte las consultas de petición necesarias, se puede utilizar el lenguaje de programación Python.

En caso de no tener instalado el lenguaje de programación, se puede proceder a instalarlo mediante el siguiente comando:

```
yay -S python
```

### Very Secure FTP Daemon (vsftpd)

Para realizar la captura de datos utilizando un servidor FTP, debemos instalar un servidor FTP compatible para sistemas operativos basados en UNIX. En este caso se optó por utilizar Very Secure FTP Daemon, el cual se puede instalar mediante el siguiente comando:

```
yay -S vsftpd
```

También se deben configurar las opciones básicas en el archivo de configuración del servidor. Se puede acceder al archivo de configuración mediante el siguiente comando:

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

Luego, se proceden a configurar las opciones básicas del servidor.

```
anonymous_enable=NO  
local_enable=YES  
write_enable=YES  
listen_port=21
```

## Un Cliente FTP

Para realizar el envío de archivos se utilizó una aplicación que permitiese ver los archivos ubicados en el servidor FTP así como también los archivos ubicados en la máquina local. En este caso se optó por utilizar FileZilla como la aplicación para enviar archivos entre la máquina local y el servidor. Para instalar FileZilla podemos usar el siguiente comando.

```
yay -S filezilla
```

## Ngrok

Ngrok es una herramienta que permite exponer un servidor local mediante túneles HTTP. La configuración e instalación de Ngrok se puede hacer mediante los siguientes comandos:

```
yay -S ngrok
```

Luego autenticamos ngrok con nuestra cuenta:

```
ngrok authtoken TU_TOKEN
```

Abrimos un túnel TCP en el puerto que configuramos para el servidor FTP (por defecto es el puerto 21)

```
ngrok tcp 21
```

## Primera Captura de Datos: Servidor HTTP

### Montar El Servidor

Debemos acceder a la ruta en la que se desea abrir el servidor para crear un directorio de datos. En este caso, la ruta elegida fue la carpeta raíz del dispositivo, es decir: `/`.

Para crear un servidor HTTP Básico usando python se puede ejecutar el módulo básico de python conocido como `http.server`. Así mismo se debe indicar el número de puerto al cual se desea asignar este servidor (En este caso se utilizará el puerto 8080). Esto se puede hacer utilizando la siguiente línea de comandos:

```
python -m http.server 8000
```

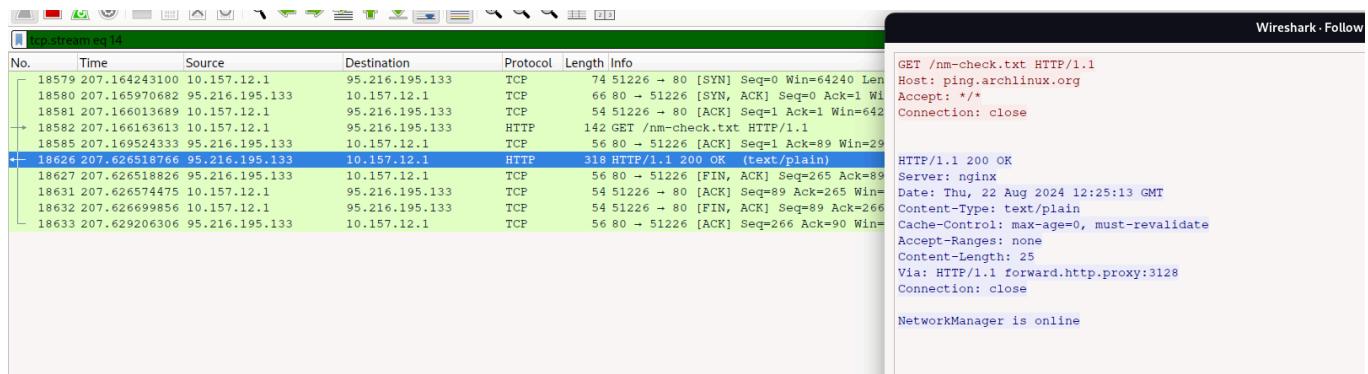
### Capturar Los Datos

Para realizar una captura de datos utilizando un Servidor HTTP Básico de Python, basta con realizar un ping a una carpeta dentro de este servidor. En este caso se realizó un ping a la carpeta raíz del servidor.

```
(base) ✘ tato@ArchBTW ~ ➔ ping localhost
PING localhost (::1) 56 data bytes
64 bytes from localhost (::1): icmp_seq=1 ttl=64 time=0.023 ms
64 bytes from localhost (::1): icmp_seq=2 ttl=64 time=0.041 ms
64 bytes from localhost (::1): icmp_seq=3 ttl=64 time=0.048 ms
64 bytes from localhost (::1): icmp_seq=4 ttl=64 time=0.088 ms
64 bytes from localhost (::1): icmp_seq=5 ttl=64 time=0.050 ms
64 bytes from localhost (::1): icmp_seq=6 ttl=64 time=0.073 ms
```

```
(base) tato@ArchBTW ➤ ~ ➤ ping localhost
PING localhost (::1) 56 data bytes
64 bytes from localhost (::1): icmp_seq=1 ttl=64 time=0.039 ms
64 bytes from localhost (::1): icmp_seq=2 ttl=64 time=0.074 ms
64 bytes from localhost (::1): icmp_seq=3 ttl=64 time=0.040 ms
64 bytes from localhost (::1): icmp_seq=4 ttl=64 time=0.057 ms
64 bytes from localhost (::1): icmp_seq=5 ttl=64 time=0.057 ms
```

Posteriormente accediendo a la tarjeta de red mediante Wireshark, se puede hacer el seguimiento al envío de paquetes mediante la selección del paquete y el seguimiento del canal HTTP:



```
GET /nm-check.txt HTTP/1.1
Host: ping.archlinux.org
Accept: /*
Connection: close

HTTP/1.1 200 OK
Server: nginx
Date: Thu, 22 Aug 2024 12:25:13 GMT
Content-Type: text/plain
Cache-Control: max-age=0, must-revalidate
Accept-Ranges: none
Content-Length: 25
Via: HTTP/1.1 forward.http.proxy:3128
Connection: close

NetworkManager is online
```

Así mismo, se procede a exportar los datos en un formato de texto plano:

The screenshot shows the Wireshark application window with a modal dialog titled "Wireshark · Export · HTTP object list". The dialog contains a table with the following data:

Packet	Hostname	Content Type	Size	Filename
18626	ping.archlinux.org	text/plain	25 bytes	nm-check.txt
164348	ping.archlinux.org	text/plain	25 bytes	nm-check.txt
292694	ping.archlinux.org	text/plain	25 bytes	nm-check.txt
423586	ping.archlinux.org	text/plain	25 bytes	nm-check.txt

Below the table are buttons for "Help", "Preview", "Save All", "Close", and "Save". To the right of the dialog, there is a vertical list of numbers: b4, d8, 39, 31, 65, 20, 34, 43, 78, 43.

## Segunda Captura: Servidor FTP

### Abrir el túnel FTP

Lo primero que tenemos que hacer es abrir un túnel TCP en el puerto configurado en el servidor FTP:

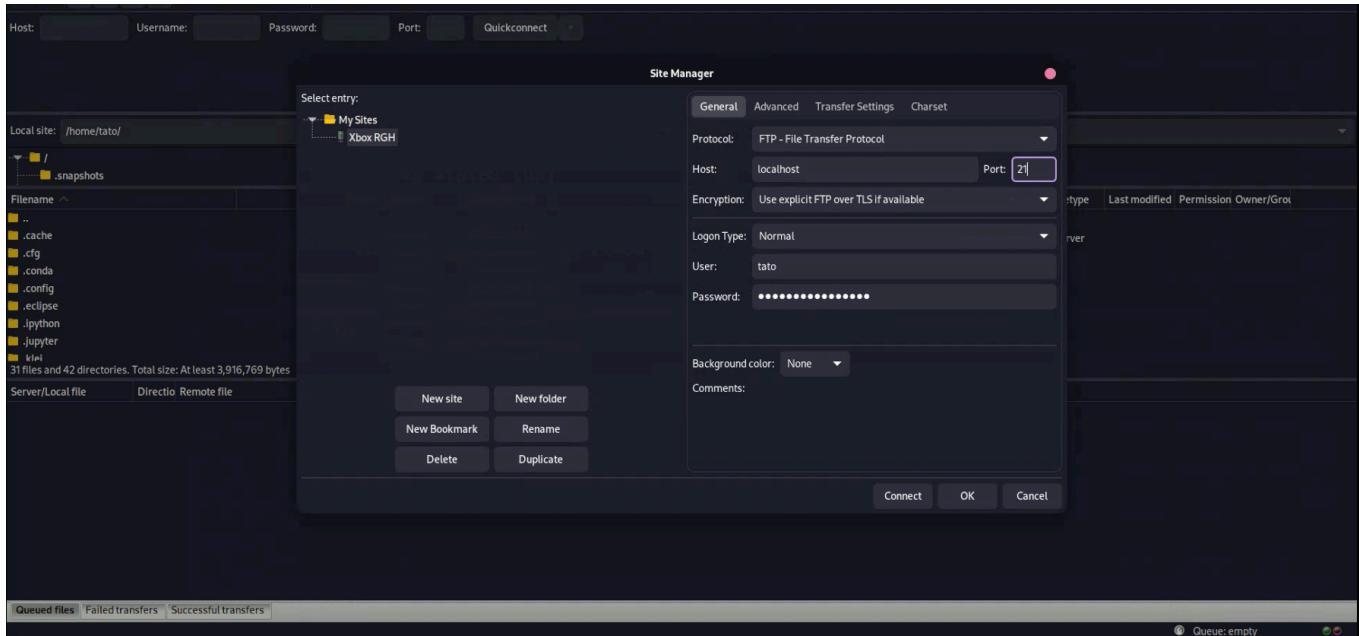
```
ngrok tcp 21
```

### Conectarse a el servidor FTP y Enviar Archivos

Posteriormente nos conectamos al servidor FTP utilizando FileZilla como el cliente FTP. Las credenciales de acceso tienen como host a tres posibles links: La interfaz web y los dos links de envío que se pueden ver de la siguiente manera

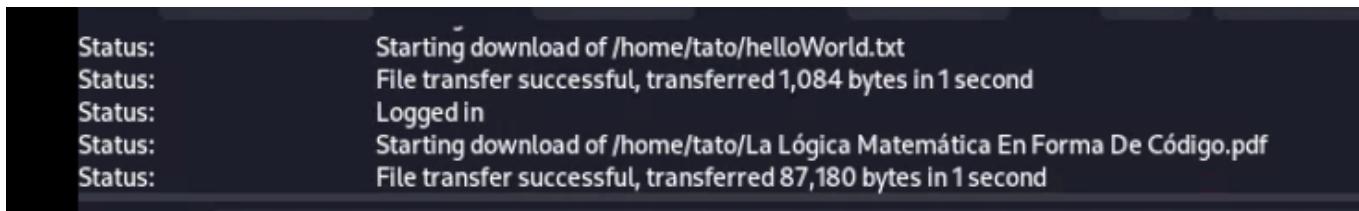
```
tcp://0.tcp.ngrok.io:XXXXX
```

Este es un ejemplo de como conectarse a el servidor usando un host llamado localhost y las credenciales del dispositivo como método de autenticación.



Luego de la conexión mediante FTP podemos enviar archivos entre el servidor FTP y la máquina local.

Este es un ejemplo de como se ve un registro de envío de archivos exitoso



## Capturar los datos

Posteriormente si accedemos a la tarjeta de red mediante Wireshark, se puede hacer el seguimiento al envío de paquetes mediante la selección del paquete y el seguimiento del canal. Para más facilidad, se decidieron filtrar los protocolos y verificar su información.

Wireshark - Network traffic analysis tool

No.	Time	Source	Destination	Protocol	Length	Info
106	12.402154880	127.0.0.1	127.0.0.1	FTP-DATA	1152	FTT Data: 1152 bytes (9216 bits) on wire (9216 bits), 1152 bytes captured (9216 bits) on interface any, id 0
128	12.4201078279	127.0.0.1	127.0.0.1	FTP-DATA	32836	FTT Data: 32836 bytes (262688 bits) on wire (262688 bits), 32836 bytes captured (262688 bits) on interface any, id 0
130	12.421114831	127.0.0.1	127.0.0.1	FTP-DATA	32836	FTT Data: 32836 bytes (262688 bits) on wire (262688 bits), 32836 bytes captured (262688 bits) on interface any, id 0
131	12.421137971	127.0.0.1	127.0.0.1	FTP-DATA	21712	FTT Data: 21712 bytes (17368 bits) on wire (17368 bits), 21712 bytes captured (17368 bits) on interface any, id 0

Context menu for selected item:

- Mark/Unmark Selected
- Ignore/Unignore Selected
- Set/Inset Time Reference
- Time Shift...
- Packet Comments
- Edit Resolved Name
- Apply as Filter
- Prepare as Filter
- Conversation Filter
- Colorize Conversation
- SCTP
- Follow
- Copy
- Protocol Preferences
- Decode As...
- Show Packet in New Window
- Analyse this Association
- Show All Associations
- Filter this Association

Packets: 217 · Displayed: 4 (1.8%) · Profile: Default

Wireshark - Network traffic analysis tool

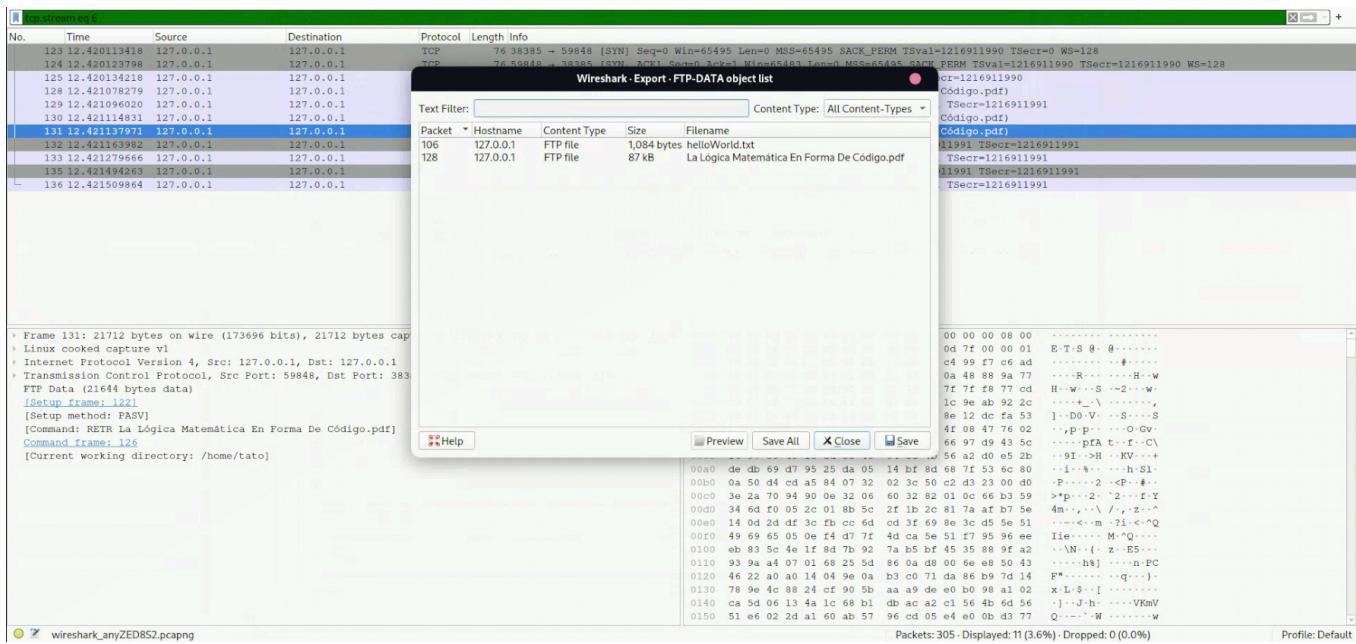
No.	Time	Source	Destination	Protocol	Length	Info
123	12.420113416	127.0.0.1	127.0.0.1	TCP	76	38385 → 59845 [SYN] Seq=0 Win=65495 SACK_PERM TSval=1216911990 TSecr=0 WS=128
124	12.4201237979	127.0.0.1	127.0.0.1	TCP	76	59848 → 38388 [SYN, ACK] Seq=1 Win=65483 Len=0 MSS=65495 SACK_PERM TSval=1216911990 TSecr=1216911990 WS=128
125	12.420134218	127.0.0.1	127.0.0.1	TCP	68	38385 → 59848 [ACK] Seq=1 Ack=1 Win=65483 Len=0 TSval=1216911990 TSecr=1216911990
128	12.421078279	127.0.0.1	127.0.0.1	FTP-DATA	32768	FTT Data: 32768 bytes (262688 bits) (RETR La Lógica Matemática En Forma De Código.pdf)
129	12.421096026	127.0.0.1	127.0.0.1	TCP	68	38385 → 59848 [ACK] Seq=1 Ack=32768 Win=65536 Len=0 TSval=1216911991 TSecr=1216911991
130	12.421114831	127.0.0.1	127.0.0.1	FTP-DATA	32764	FTT Data: 32764 bytes (262684 bits) (RETR La Lógica Matemática En Forma De Código.pdf)
131	12.421137971	127.0.0.1	127.0.0.1	FTP-DATA	21712	FTT Data: 21712 bytes (17368 bits) (RETR La Lógica Matemática En Forma De Código.pdf)
132	12.421163986	127.0.0.1	127.0.0.1	TCP	68	39848 → 38385 [FIN, ACK] Seq=87181 Ack=1 Win=65536 Len=0 TSval=1216911991 TSecr=1216911991
133	12.421279666	127.0.0.1	127.0.0.1	TCP	68	38385 → 59848 [ACK] Seq=1 Ack=87182 Win=65536 Len=0 TSval=1216911991 TSecr=1216911991
135	12.421494263	127.0.0.1	127.0.0.1	TCP	68	38385 → 59848 [FIN, ACK] Seq=1 Ack=87182 Win=65536 Len=0 TSval=1216911991 TSecr=1216911991
136	12.421509864	127.0.0.1	127.0.0.1	TCP	68	59848 → 38385 [ACK] Seq=87182 Ack=2 Win=65536 Len=0 TSval=1216911991 TSecr=1216911991

Context menu for selected item:

- Mark/Unmark Selected
- Ignore/Unignore Selected
- Set/Inset Time Reference
- Time Shift...
- Packet Comments
- Edit Resolved Name
- Apply as Filter
- Prepare as Filter
- Conversation Filter
- Colorize Conversation
- SCTP
- Follow
- Copy
- Protocol Preferences
- Decode As...
- Show Packet in New Window
- Analyse this Association
- Show All Associations
- Filter this Association

Packets: 305 · Displayed: 11 (3.6%) · Dropped: 0 (0.0%) · Profile: Default

Después podemos exportar los archivos mediante la descarga de todos los datos de tipo FTP-DATA



## Conclusión

Este laboratorio ha demostrado cómo la captura y el análisis de datos en tránsito pueden revelar detalles importantes sobre la seguridad y el comportamiento de los protocolos de comunicación HTTP y FTP. La configuración y utilización de herramientas como `nginx`, `vsftpd`, `ngrok`, y Wireshark han permitido no solo observar las diferencias estructurales entre estos protocolos, sino también evidenciar las posibles vulnerabilidades que cada uno presenta en un entorno controlado.

A través de la captura de datos HTTP, se pudo identificar la exposición directa de información durante las solicitudes no protegidas, lo que resalta la importancia de implementar medidas de seguridad adicionales en aplicaciones web. En el caso del FTP, la captura de datos mostró cómo los archivos transmitidos pueden ser interceptados, subrayando la necesidad de utilizar protocolos más seguros o añadir capas de cifrado para proteger la integridad y confidencialidad de los datos.

Este ejercicio no solo contribuye al entendimiento teórico de estos protocolos, sino que también ofrece una experiencia práctica valiosa para la gestión y administración de redes, permitiendo a los profesionales identificar y mitigar riesgos en escenarios reales de seguridad informática.