NANOFILES: DOCUMENTACIÓN DE DISEÑO

(1904) REDES DE COMUNICACIONES [23/24]
Profesor: Epifanio Gaona Ramírez

Descripción breve

Documento entregable con información útil para la implementación, entendimiento y evaluación de la tarea práctica de la asignatura Redes de Comunicaciones

Álvaro Aledo Tornero Y Antonio Vergara Moya
48129547T y 48851755P

Índice

Introducción

En este documento se especifica el diseño de los protocolos TCP y UDP para la comunicación entre procesos *Nanofiles* entre ellos mismos y un proceso servidor *Directory*. Además, se especifican y explican las mejoras adicionales que han sido añadidas a ambos programas y se muestran fragmentos de comunicaciones rastreadas con Wireshark.

Formato de los mensajes del protocolo de comunicación con el Directorio

Para definir el protocolo de comunicación con el *Directorio*, vamos a utilizar mensajes textuales con formato "campo:valor" que implementaremos mediante la programación de la clase DirMessage. El valor que tome el campo "operation" (código de operación) indicará el tipo de mensaje y por tanto su formato (qué campos vienen a continuación).

Mensajes de solicitud

login

Descripción: Este mensaje lo envía el cliente de NanoFiles al Directorio para solicitar "iniciar sesión" y registrar el nickname deseado indicado en el mensaje.
Ejemplo:
operation: login\n
nickname: alicia\n
\n

logout

Descripción: Este mensaje lo usa el cliente NanoFiles con una sesión iniciada para indicarle al Directorio que quiere cerrar su sesión actual.
Ejemplo:
operation: logout\n
sessionKey: 1234\n
\n

userlist

Descripción: Mensaje disponible para los clientes NanoFiles que solita al Directorio la lista de los usuarios que están conectados e información de sus puertos si son servidores y del tipo de usuario que son en el momento del procesamiento de la solicitud.

Ejemplo:
operation: userlist\n
sessionKey: 1234\n
\n

publish

Descripción: Mensaje disponible para los clientes NanoFiles que son servidores que les permite publicar (y en un principio actualizar) los ficheros que tiene disponibles para compartir. De sus ficheros se guardarán en el Directorio su nombre, hash y tamaño en bytes. Este mensaje se intenta ejecutar automáticamente cuando se crea el servidor.
Ejemplo:
operation: publish\n
sessionKey: 134\n
files: fileHash1,fileName1,fileSize1:fileHash2,fileName2,...\n
\n

filelist

Descripción: Mensaje que usan los clientes NanoFiles para solicitarle al Directorio un listado de los ficheros publicados que contenga su nombre, hash y tamaño en bytes.
Ejemplo:
operation: filelist\n
sessionKey: 1324\n
\n

search

Descripción: Mensaje que usan los clientes NanoFiles para preguntarle al Directorio qué servidores de los registrados comparten un fichero con un hash completo específico.
Ejemplo:
operation: search\n
sessionKey: 145\n
hash: abc123\n
\n

downloadfrom

Descripción: Mensaje que se usan los clientes NanoFiles en ciertas solicitudes de descarga cuando lo intentan hacer solo con el Nick del servidor. Este mensaje se manda al Directorio y el busca a dicho servidor para intentar dar la IP y el puerto de este.
Ejemplo:
operation: downloadfrom\n
sessionKey: 145\n
nick: server\n
\n

register_server

Descripción: Mensaje que usan automáticamente al iniciarse los servidores Nanofiles que han iniciado sesión en un Directorio. Este mensaje se usa para registrar la creación del servidor y la información necesaria para comunicarse con él (IP y puerto).
Ejemplo:
operation: register_server\n
sessionKey: 145\n
port: 23\n
\n

unregister

Descripción: Mensaje que usan automáticamente los servidores Nanofiles registrados en un Directorio. Es usado para que el Directorio deje de considerar este servidor como registrado y elimine a su vez toda traza de los ficheros que tenía publicados.
Ejemplo:
operation: unregister_server\n
sessionKey: 1324\n
\n

Mensajes de respuesta

loginok

Descripción: Este mensaje lo manda el Directorio en respuesta a un mensaje de operación login cuando el login ha sido exitoso, adjuntándole la sessionKey correspondiente al usuario.
Ejemplo:
operation: loginok\n
sessionKey: 1234 \n
\n

login_failed

Descripción: Este mensaje se manda cuando la operación de login no se ha podido realizar. Este mensaje se recibe cuando el nombre de usuario que se ha querido usar ya se encuentra en uso.
Ejemplo:
operation: login_failed\n
\n

logoutok

Descripción: Este mensaje se manda cuando se ha cerrado sesión exitosamente.
Ejemplo:
operation: logoutok\n
\n

logout_failed

Descripción: Este mensaje se manda cuando no se ha podido cerrar sesión. Se manda si alguien consigue contactar con el directorio sin estar logeado en él o si tiene un servidor registrado en el mismo.
Ejemplo:
operation: logout_failed\n
\n

userlistok

Descripción: Este mensaje se usa para mandar la lista de usuarios al cliente nanofiles. En esta lista se menciona, el nombre de los usuarios, si son usuarios normales o servidores y, en caso de ser servidores, en qué puerto escuchan.
Ejemplo:
operation: userlistok\n
users: "usuario1:tipo1:[puerto1], usuario2:tipo2:[puerto2],..."\n
\n

userlist_failed

Descripción: Este mensaje lo manda Directory para indicar al usuario que ha fallado la operación de userlist.
Ejemplo:
operation: userlist_failed\n
\n

publishok

Descripción: Este mensaje lo manda Directory para indicar al servidor NanoFiles que ha sido registrado exitosamente.
Ejemplo:
operation: publish_ok\n
\n

publish_failed

Descripción: Este mensaje lo manda Directory para indicar que el servidor NanoFiles no se puede registrar.
Ejemplo:
operation: publish_failed\n
\n

filelistok

Description: Este mensaje lo manda el Directorio para mandar la lista de ficheros disponibles.
Ejemplo:
files:
file1Hash,file1Name,file1Size,fileServer1/fileServer2:file2Hash,file2Name,
...\n
sessionKey: 1324\n

\n

filelist_failed

Descripción: Este mensaje lo manda el Directorio para indicar que no se ha podido dar la lista de ficheros.
Ejemplo:
operation: filelist_failed\n
\n

searchok

Descripción: Este mensaje lo manda el Directorio para dar los servidores que comparten un archivo en concreto.
Ejemplo:
operation: searchok\n
servers: server1,server2,server3,...\n
\n

search_failed

Descripción: Este mensaje lo manda el Directorio para indicar que no se han encontrado servidores que compartan el archivo deseado.
Ejemplo:
operation: search_failed\n
\n

downloadfromok

Descripción: Este mensaje lo manda el Directorio para dar la información asociada al nick que se ha dado para descargar.
Ejemplo:
operation: downloadformok\n
Ip: 0.0.0.0\n
Port: 1\n
\n

downloadfrom_failed

Descripción: Este mensaje lo manda el Directorio para indicar que no se ha podido encontrar ningún servidor registrado
Ejemplo:
operation: downloadfrom_failed\n
\n

register_serverok

Descripción: Este mensaje lo manda el Directorio para indicar que se ha registrado exitosamente al servidor.
Ejemplo:
operation: register_serverok\n
\n

register_server_failed

Descripción: Este mensaje lo manda el Directorio para indicar que ha habido un fallo a la hora de registrar el servidor.
Ejemplo:
operation: register_server_failed\n
\n

unregister_serverok

Descripción: Este mensaje de Directorio indica que el servidor y sus ficheros publicados han sido quitados de los registros existosamente.
Ejemplo:
operation: unregister_serverok\n
\n

unregister_server_failed

Descripción: Este mensaje de Directorio indica que el servidor y sus ficheros no se ha podido quitar de los registros.
Ejemplo:
operation: unregister_server_failed\n
\n

Formato de los mensajes del protocolo de transferencia de ficheros

Para definir el protocolo de comunicación con un servidor de ficheros, vamos a utilizar mensajes binarios multiformato. El valor que tome el campo “opcode” (código de operación) indicará el tipo de mensaje y por tanto cuál es su formato, es decir, qué campos vienen a continuación.

Tipos y descripción de los mensajes

Mensaje: **FILE_NOT_FOUND (opcode = 1)**
Sentido de la comunicación: Servidor de ficheros → Cliente
Descripción: Este mensaje lo envía el par servidor de ficheros al par cliente (receptor) de fichero para indicar que no es posible encontrar el fichero con la información proporcionada en el mensaje de petición de descarga.
Ejemplo:

Opcode (1 byte)
1

Mensaje**: DOWNLOAD_FROM** (opcode = 2)
Sentido de la comunicación: Cliente -> Servidor de ficheros
Descripción: Este mensaje se usa para descargar un fichero dado su hash (o una subcadena del mismo)

Ejemplo:

Opcode **(1 byte)**	**HashFile** **(4 bytes)**
2	**1234**

Mensaje: **DOWNLOAD_OK** (opcode = 3)
Sentido de la comunicación: Servidor de ficheros -> Cliente
Descripción: Este mensaje se usa para indicar que la solicitud de descarga ha sido aceptada. Tiene adjuntado el fichero.

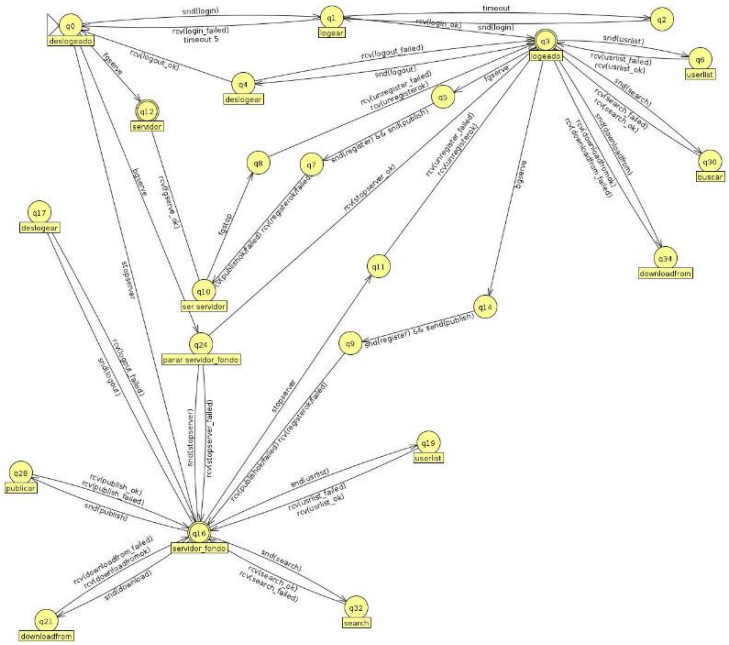
Opcode **(1 byte)**	**fileLength** **(4 bytes)**	**File** **(fileLength bytes)**
3	**5**	**file**

Mensaje**: AMBIGUOUS_DOWNLOAD** (opcode = 4)
Sentido de la comunicación: Cliente -> Servidor de ficheros
Descripción: Este mensaje se usa para mandar los nombres de los ficheros y sus hashes completos a los que se podía referir el subhash mandado.
Ejemplo:

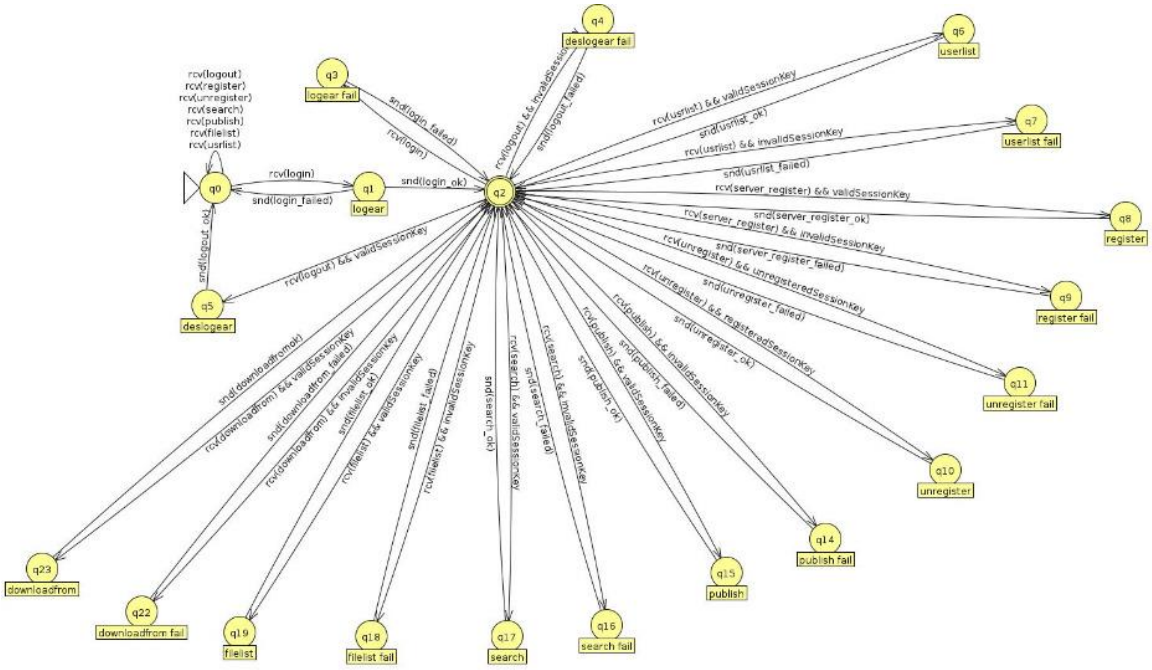
Opcode **(1 byte)**	**HashFile** **(1-n bytes)**	**FileName** **(1-m bytes)**
4	**hash1:hash2:hash3:...**	**name1:name2:name3:...**

Autómatas de protocolo

Autómata rol Nanofiles

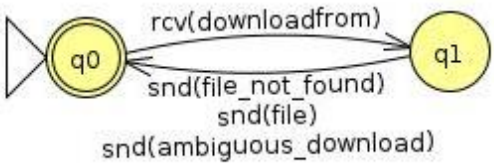


Autómata rol Directorio



Autómata rol Cliente P2P

Autómata rol Servidor P2P



Mejoras implementadas

Fgserve puerto variable: 0.5 punto

Esta mejora es bastante sencilla de implementar. Al preparar el servidor (construir la instancia de NFServerSimple) hacemos uso de una variable auxiliar para el puerto que se intenta usar. Esta variable inicialmente se inicia al valor del puerto predeterminado y se usa para crear el serverSocketAddress. Si la construcción falla, (el puerto que se ha intentado usar está ocupado) simplemente aumentamos la variable en uno. Continuamos intentándolo con este método hasta que encontremos un puerto disponible.

Downloadfrom por nickname: 1 punto

Esta mejora es fundamental para el funcionamiento del resto de mejoras ya que comparte una gran parte del código con muchas otras. Esta mejora tiene tres partes, la parte del cliente, la parte del servidor y la parte del directorio:

Para el cliente, éste tiene que comprobar si los parámetros del comando downloadfrom son de la forma de IP:puerto o no. Si lo es, esta mejora no se usa ya que no es necesario. En caso contrario, si el cliente ha iniciado sesión en el Directorio, hará una solicitud al Directorio para que le diga la IP y el puerto asociado a ese nickname. Si la respuesta es exitosa el cliente recibirá la información y podrá hacer la descarga, en caso contrario, no podrá realizarla.

Para el servidor, lo único que tiene que hacer es intentar mandar una solicitud al Directorio cuando inicie el servidor para indicarle que existe y pasarle su puerto (su ip ya la tendrá por el mero hecho de haber recibido su mensaje). Si esto resulta exitoso, el servidor estará registrado en el Directorio y su información podrá ser compartida. Si no, el servidor se creará, pero no habrá forma de comunicarse por él si no se sabe de antemano su IP y su puerto.

En la parte del directorio se crean dos estructuras de dato tipo HashMap que asocian las sessionKeys de los servidores a sus IPs y puertos. Si un usuario logeado se registra como servidor, sus datos quedarán guardados en esas estructuras. Cuando el Directorio reciba solicitudes de descarga comprobará si ese servidor está registrado y pasará sus datos si los encuentra.

Bgserve secuencial: 1 punto

Para crear esta mejora, se han seguido los TODOs de la clase NFCControllerLogicP2P y se ha implementado y ejecutado la clase NFServer en un hilo secundario. De esta manera se pueden usar comandos y recibir solicitudes de descarga.

Bgserve multihilo: 0.5 punto

Para hacer que el servidor de segundo plano sea multihilo, se ha seguido la información del boletín que hablaba de su implementación. Se ha modificado la clase NFServer para que a cada solicitud cree un hilo que ejecute la clase NFServerThread para responder a las múltiples solicitudes.

Stopserver: 0.5 punto

Para implementar esta mejora, en cada servidor de primer plano se establece un bufferedReader (un escaner no funciona, ya que da conflictos con el que se usa para leer normalmente los comandos) y se usa para ver si se ha escrito el comando fgstop. Si es el caso el servidor se para, si no, continua. En el servidor de segundo plano, como seguimos disponiendo del terminal, no es necesario implementar nada nuevo aparte de la funcionalidad del comando stopserver. Para hacer la comprobación de que se ha escrito el comando deseado y dejar de escuchar solicitudes. Se le establece al socket un timeout que pare cada segundo la escucha.

Bgserve puerto efímero: 0.5 punto

Siguiendo la propia información que da la documentación de java, si a un InetSocketAddress a la hora de crearlo le pasas un 0 como parámetro, se usa un puerto efímero.

Userlist ampliado con servidores: 0.5 punto

Para mejorar el comando userlist, a nivel de usuario nos hemos basado en la impresión formateada de ficheros que hay en la clase FileInfo para imprimir toda la información y hemos modificado la respuesta a userlist haciendo que el mensaje contenga el tipo de usuario asociado a cada nick y, si ese nick es un servidor, su puerto. Esta mejora depende de que los servidores se registren.

Publish + filelist: 0.5 punto

Para implementar esta mejora, se hizo que, con cada iniciación de servidor, además de registrarse, publicara sus ficheros. El servidor manda en un mensaje para el Directorio en el que para cada fichero se da su nombre, hash y tamaño. Cuando el Directorio lo recibe, si el servidor está registrado, éste los guardará en un HashMap con la estructura de <sessionKey, <hash, [name, size]>>. También, por si acaso fuera una actualización por parte de un servidor en segundo plano (aunque esto no hace nada, porque las bases de datos no se actualizan en Nanofiles), el Directorio elimina cualquier fichero publicado por ese servidor antes de añadir los que se han mandado.

A la hora de solicitar una lista de ficheros, su impresión se ha dejado en manos del método ya creado en FileInfo (como se pide en los TODOs). Y para mandarla se hace uso de una string con patrón: hash,name,size:hash,name,size:... para que pueda ser interpretada fácilmente por el usuario. También, se usa una estructura Set para garantizar que ningún archivo compartido por varios servidores sea repetido.

Publish + search: 0.5 punto

Con la estructura que tenemos implementada en el Directorio esta operación es bastante sencilla, recorremos todas las sessionsKeys asociadas a ficheros publicados, si uno de sus hashes coincide con el dado, se busca el nick del servidor y se añade al mensaje de respuesta. Si la cadena estuviera vacía, significa que la búsqueda ha fallado porque no se ha encontrado el archivo en ninguno de los servidores registrados.

Filelist ampliado con servidores : 0.5 punto

Para realizar esta mejora, simplemente se cambia la codificación del mensaje de respuesta a hash,name,size,server1/server2:hash,name,... y se modifica la extracción de datos por parte del cliente para poder interpretar bien la solución. Para obtener los nicks de los servidores, simplemente se asocian en un HashMap temporal los hashes de los ficheros con los servidores que lo publican y se construye la repuesta.

Baja ficheros y servidores: 0.5 punto

Como esta mejora solo se aplica cuando se cierran servidores, esta mejora es tan sencilla como hacer que cuando se cierre un servidor, si éste ha sido registrado que mande un mensaje solicitando que se le elimine de los registros. Cuando el Directorio lo reciba, si la sessionKey es de un servidor, eliminará todas las entradas con esa clave en las tablas de ficheros y de IPs y puertos.

Ejemplo de intercambio de mensajes

CLIENTE1: inicia sesión
Operation: login
Nickname: alumno

DIRECTORIO: acepta inicio de sesión
operation: loginok
userkey: 123

CLIENTE2: inicia sesión
Operation: login
Nickname: alumno

DIRECTORIO: deniega inicio de sesión
Operation: login_failed

CLIENTE1: pide lista de usuarios
Operation: userlist
Userkey: 123

DIRECTORIO: da lista de usuario
Operation: userlistok
Users: alumno:User,

CLIENTE2: inicia sesión
Operation: login
Nickname: pepe

DIRECTORIO: acepta inicio de sesión
Operation: loginok
Userkey: 145

CLIENTE1: pide lista de usuarios
Operation: userlist
Userkey: 123

DIRECTORIO: da lista de usuarios
Operation: userlistok
Users: alumno:User,pepe:User,

CLIENTE2: se hace servidor en segundo plano (se registra y publica archivos)
Operation: register_server
Userkey: 145
Port: 1894

Operation: publish
Userkey: 145
Files: fichero1Hash,fichero1Nombre,fichero1Tamaño:fichero2Hash,...

DIRECTORIO registra servidor y sus ficheros
Operation: register_serverok

Operation: publishok

CLIENTE1: pide lista de usuarios
Operation: userlist
Userkey: 123

DIRECTORIO: da lista de usuarios
Operation: userlistok
Users: alumno:User,pepe:Server Port: 1894,

CLIENTE2: pide lista de ficheros
Operation: filelist
Userkey: 145

DIRECTORIO: da lista de ficheros
Operation: filelistok
Files: fichero1Hash,fichero1Nombre,fichero1Tamaño,pepe:fichero2Hash,...

CLIENTE1: pide cierre de sesión
Operation: logout
Userkey: 123

DIRECTORIO: cierra sesión de CLIENTE1
Operation:logoutok

CLIENTE2: cierra servidor (pide quitarse de registros y despublicar sus ficheros)

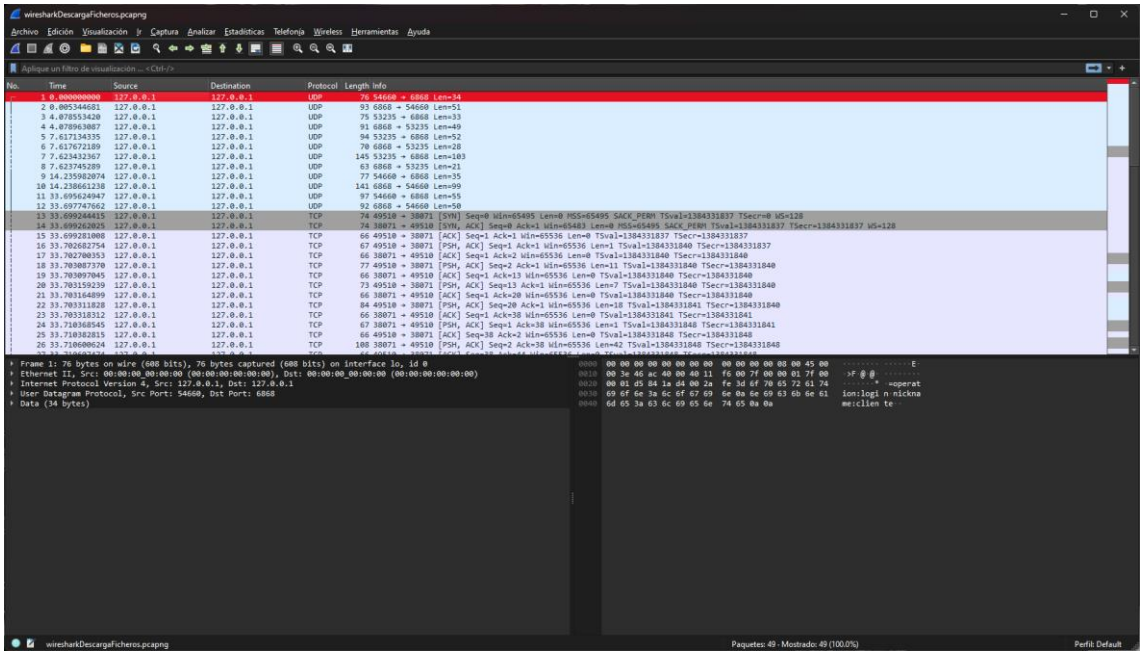
Operation: unregister_server
Userkey: 145

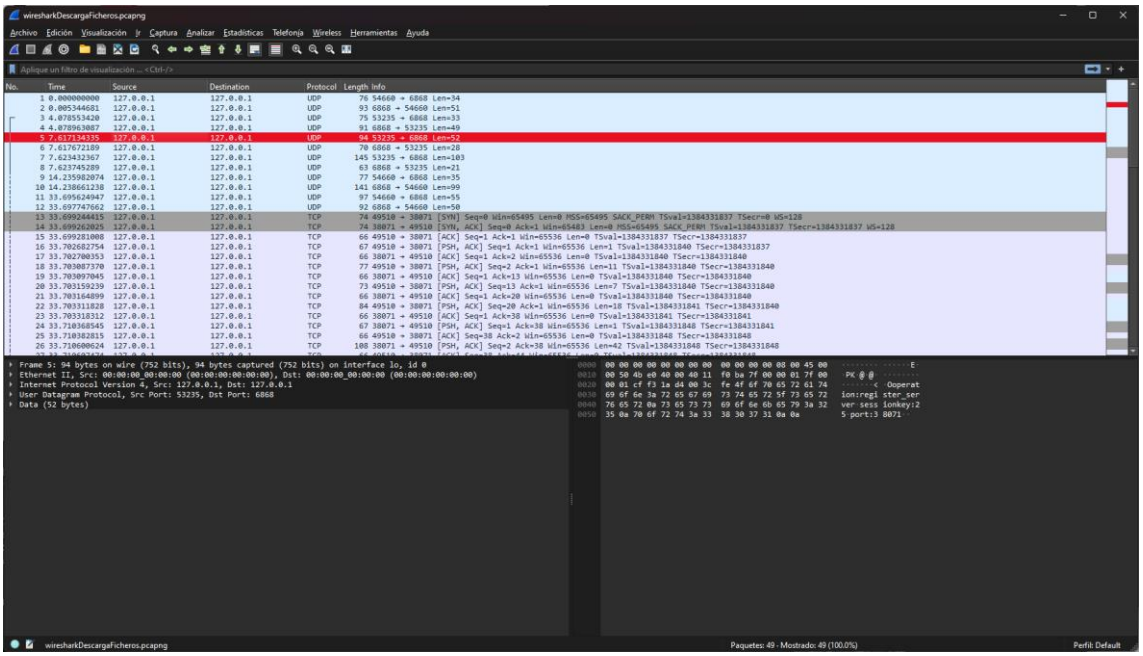
DIRECTORIO: desregistra CLIENTE2
Operation: unregister_serverok

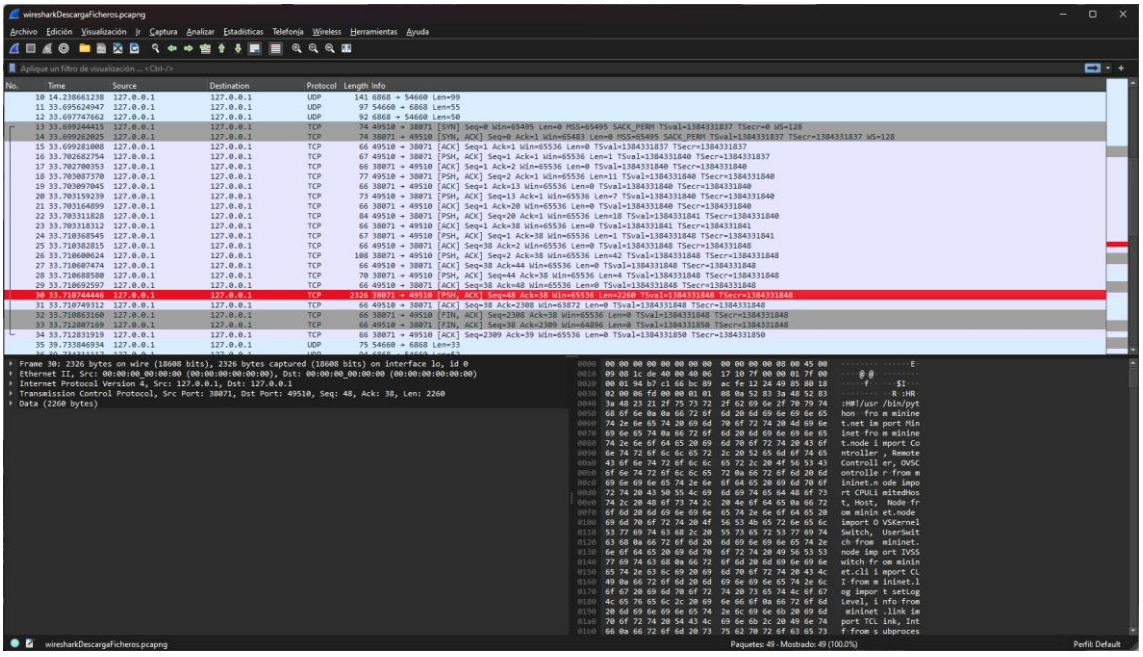
CLIENTE1: pide cierre de sesión
Operation: logout
Userkey: 145

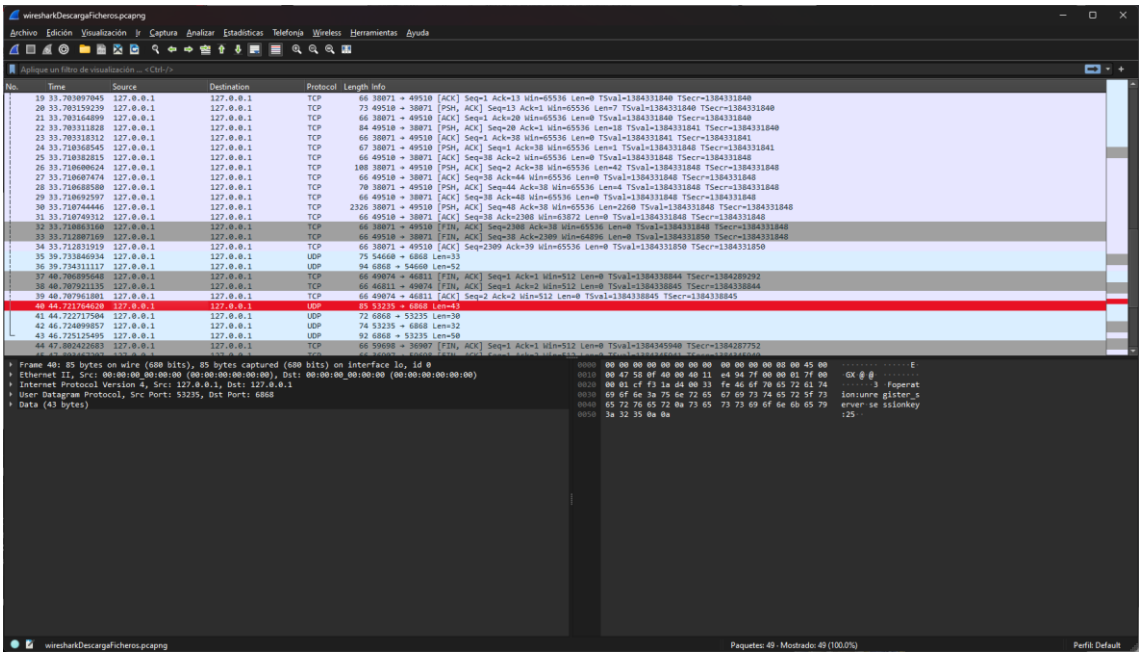
DIRECTORIO: cierra sesión de CLIENTE1
Operation:logoutok

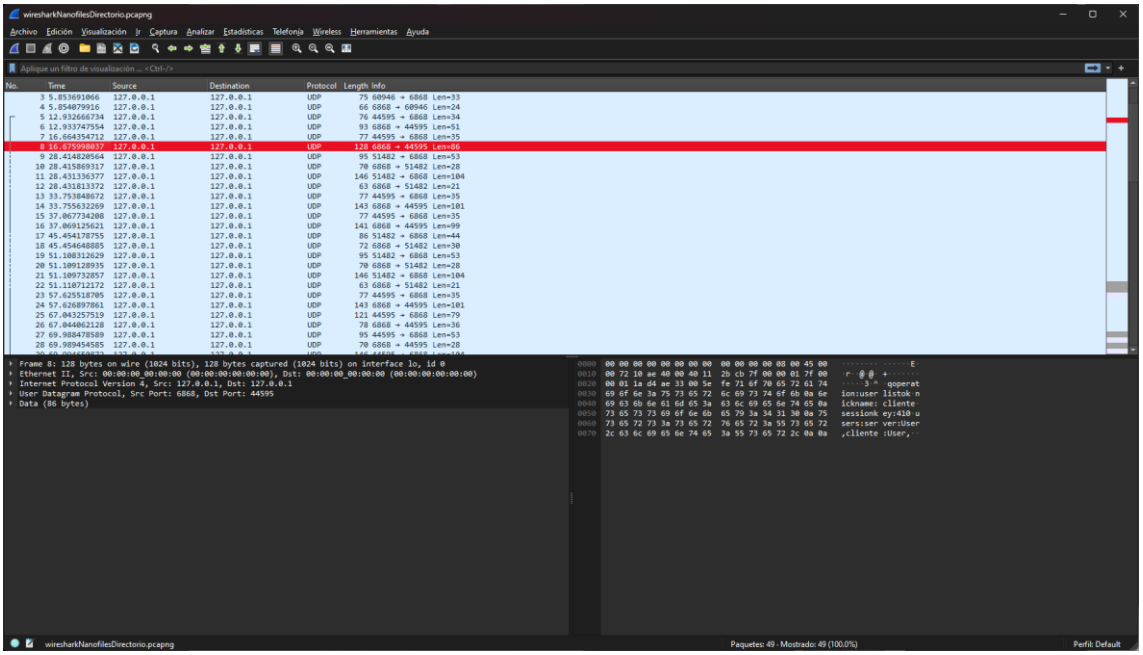
Capturas de ejecuciones

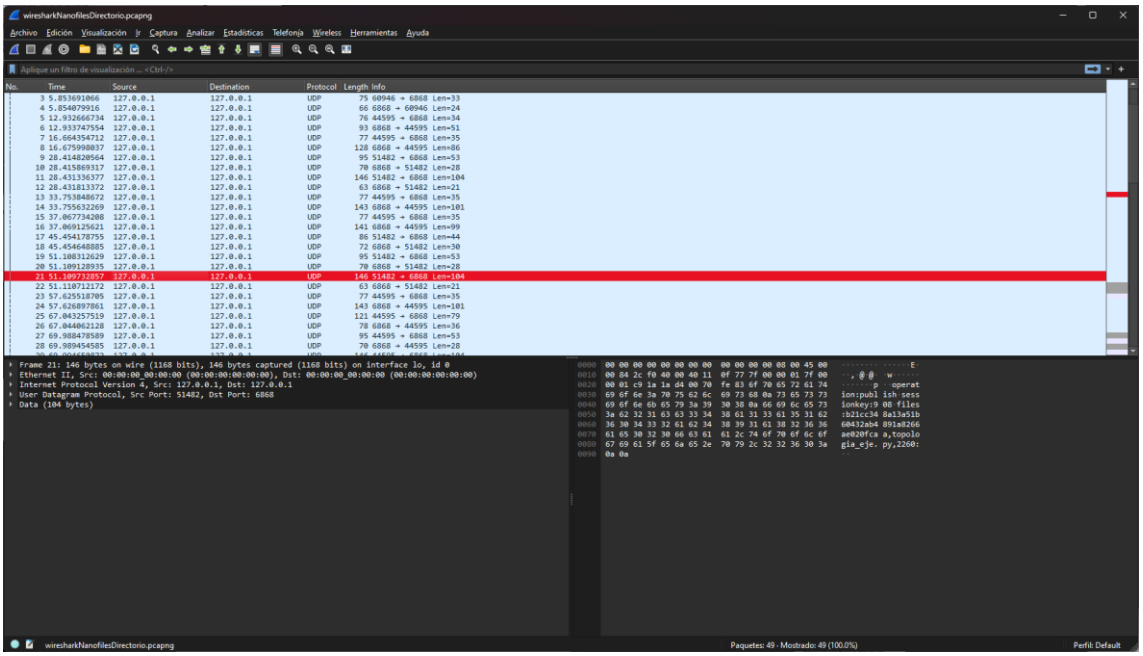


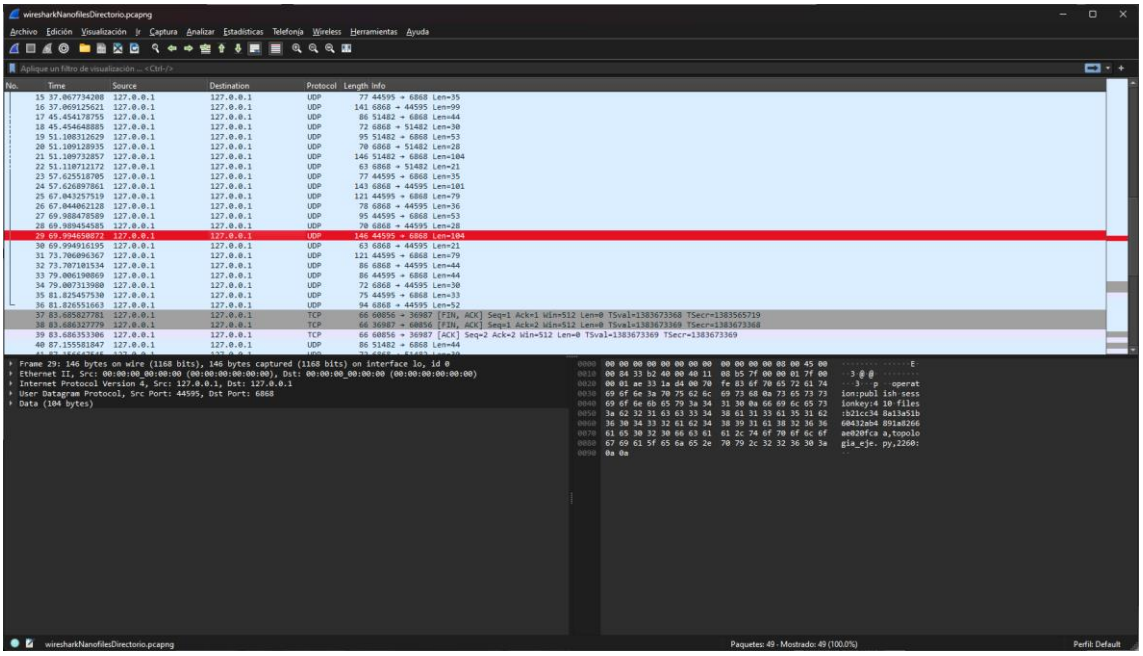

wiresharkDescargaFicheros.pcapng
Archivo Edición Visualización Ir Captura Analizar Estadísticas Telefonía Wireless Herramientas Ayuda
Aplique un filtro de visualización ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
1 0.000000000 127.0.0.1 127.0.0.1 UDP 76 54660 → 6868 Len=34
2 0.005344681 127.0.0.1 127.0.0.1 UDP 93 6868 → 54660 Len=51
3 4.078553420 127.0.0.1 127.0.0.1 UDP 75 53235 → 6868 Len=33
4 4.078963087 127.0.0.1 127.0.0.1 UDP 91 6868 → 53235 Len=49
5 7.617134335 127.0.0.1 127.0.0.1 UDP 94 53235 → 6868 Len=52
6 7.617672189 127.0.0.1 127.0.0.1 UDP 70 6868 → 53235 Len=28
7 7.623432367 127.0.0.1 127.0.0.1 UDP 145 53235 → 6868 Len=103
8 7.623745289 127.0.0.1 127.0.0.1 UDP 63 6868 → 53235 Len=21
9 14.235982074 127.0.0.1 127.0.0.1 UDP 77 54660 → 6868 Len=35
10 14.238661238 127.0.0.1 127.0.0.1 UDP 141 6868 → 54660 Len=99
11 33.695624947 127.0.0.1 127.0.0.1 UDP 97 54660 → 6868 Len=55
12 33.697747662 127.0.0.1 127.0.0.1 UDP 92 6868 → 54660 Len=50
13 33.699244415 127.0.0.1 127.0.0.1 TCP 74 49510 → 38071 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1384331837 TSecr=0 WS=128
14 33.699262025 127.0.0.1 127.0.0.1 TCP 74 38071 → 49510 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM TSval=1384331837 TSecr=1384331837 WS=128
15 33.699281008 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1384331837 TSecr=1384331837
16 33.702682754 127.0.0.1 127.0.0.1 TCP 67 49510 → 38071 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=1 TSval=1384331840 TSecr=1384331837
17 33.702700353 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=2 Win=65536 Len=0 TSval=1384331840 TSecr=1384331840
18 33.703087370 127.0.0.1 127.0.0.1 TCP 77 49510 → 38071 [PSH, ACK] Seq=2 Ack=1 Win=65536 Len=11 TSval=1384331840 TSecr=1384331840
19 33.703097045 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=13 Win=65536 Len=0 TSval=1384331840 TSecr=1384331840
20 33.703159239 127.0.0.1 127.0.0.1 TCP 73 49510 → 38071 [PSH, ACK] Seq=13 Ack=1 Win=65536 Len=7 TSval=1384331840 TSecr=1384331840
21 33.703164899 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=20 Win=65536 Len=0 TSval=1384331840 TSecr=1384331840
22 33.703311828 127.0.0.1 127.0.0.1 TCP 84 49510 → 38071 [PSH, ACK] Seq=20 Ack=1 Win=65536 Len=18 TSval=1384331841 TSecr=1384331840
23 33.703318312 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=38 Win=65536 Len=0 TSval=1384331841 TSecr=1384331841
24 33.710368545 127.0.0.1 127.0.0.1 TCP 67 38071 → 49510 [PSH, ACK] Seq=1 Ack=38 Win=65536 Len=1 TSval=1384331848 TSecr=1384331841
25 33.710382815 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=2 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
26 33.710600624 127.0.0.1 127.0.0.1 TCP 108 38071 → 49510 [PSH, ACK] Seq=2 Ack=38 Win=65536 Len=42 TSval=1384331848 TSecr=1384331848
27 33.710607474 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=44 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
28 33.710688580 127.0.0.1 127.0.0.1 TCP 70 38071 → 49510 [PSH, ACK] Seq=44 Ack=38 Win=65536 Len=4 TSval=1384331848 TSecr=1384331848
29 33.710692597 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=48 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
30 33.710744446 127.0.0.1 127.0.0.1 TCP 2326 38071 → 49510 [PSH, ACK] Seq=48 Ack=38 Win=65536 Len=2260 TSval=1384331848 TSecr=1384331848
31 33.710749312 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=2308 Win=63872 Len=0 TSval=1384331848 TSecr=1384331848
32 33.710863160 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [FIN, ACK] Seq=2308 Ack=38 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
33 33.712807160 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [FIN, ACK] Seq=38 Ack=2309 Win=64896 Len=0 TSval=1384331850 TSecr=1384331848
34 33.712831919 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=2309 Ack=39 Win=65536 Len=0 TSval=1384331850 TSecr=1384331850
35 39.733846934 127.0.0.1 127.0.0.1 UDP 75 54660 → 6868 Len=33
Frame 5: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 53235, Dst Port: 6868
Data (52 bytes)
0000 00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00 ..E.
0010 00 50 4b e0 40 00 40 11 f0 ba 7f 00 00 01 7f 00 .PK.@.@.
0020 00 01 cf f3 1a d4 00 3c fe 4f 6f 70 65 72 61 74 .<.Operat
0030 69 6f 6e 3a 72 65 67 69 73 74 65 72 5f 73 65 72 ion:regi ster_ser
0040 76 65 72 0a 73 65 73 73 69 6f 6e 6b 65 79 3a 32 ver ses lonkey:2
0050 35 0a 70 6f 72 74 3a 33 38 30 37 31 0a 0a 5 port:3 8071
wiresharkDescargaFicheros.pcapng
Paquetes: 49 · Mostrado: 49 (100.0%)
Perfil: Default


wiresharkDescargaFicheros.pcapng
Archivo Edición Visualización Ir Captura Analizar Estadísticas Telefonía Wireless Herramientas Ayuda
Aplique un filtro de visualización ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
10 14.238661238 127.0.0.1 127.0.0.1 UDP 141 6868 → 54660 Len=99
11 33.695624947 127.0.0.1 127.0.0.1 UDP 97 54660 → 6868 Len=55
12 33.697747662 127.0.0.1 127.0.0.1 UDP 92 6868 → 54660 Len=50
13 33.699244415 127.0.0.1 127.0.0.1 TCP 74 49510 → 38071 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1384331837 TSecr=0 WS=128
14 33.699262025 127.0.0.1 127.0.0.1 TCP 74 38071 → 49510 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM TSval=1384331837 TSecr=1384331837 WS=128
15 33.699281008 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1384331837 TSecr=1384331837
16 33.702682754 127.0.0.1 127.0.0.1 TCP 67 49510 → 38071 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=1 TSval=1384331840 TSecr=1384331837
17 33.702700353 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=2 Win=65536 Len=0 TSval=1384331840 TSecr=1384331840
18 33.703087370 127.0.0.1 127.0.0.1 TCP 77 49510 → 38071 [PSH, ACK] Seq=2 Ack=1 Win=65536 Len=11 TSval=1384331840 TSecr=1384331840
19 33.703097045 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=13 Win=65536 Len=0 TSval=1384331840 TSecr=1384331840
20 33.703159239 127.0.0.1 127.0.0.1 TCP 73 49510 → 38071 [PSH, ACK] Seq=13 Ack=1 Win=65536 Len=7 TSval=1384331840 TSecr=1384331840
21 33.703164899 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=20 Win=65536 Len=0 TSval=1384331840 TSecr=1384331840
22 33.703311828 127.0.0.1 127.0.0.1 TCP 84 49510 → 38071 [PSH, ACK] Seq=20 Ack=1 Win=65536 Len=18 TSval=1384331841 TSecr=1384331840
23 33.703318312 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=38 Win=65536 Len=0 TSval=1384331841 TSecr=1384331841
24 33.710368545 127.0.0.1 127.0.0.1 TCP 67 38071 → 49510 [PSH, ACK] Seq=1 Ack=38 Win=65536 Len=1 TSval=1384331848 TSecr=1384331841
25 33.710382815 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=2 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
26 33.710600624 127.0.0.1 127.0.0.1 TCP 108 38071 → 49510 [PSH, ACK] Seq=2 Ack=38 Win=65536 Len=42 TSval=1384331848 TSecr=1384331848
27 33.710607474 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=44 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
28 33.710688580 127.0.0.1 127.0.0.1 TCP 70 38071 → 49510 [PSH, ACK] Seq=44 Ack=38 Win=65536 Len=4 TSval=1384331848 TSecr=1384331848
29 33.710692597 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=48 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
30 33.710744446 127.0.0.1 127.0.0.1 TCP 2326 38071 → 49510 [PSH, ACK] Seq=48 Ack=38 Win=65536 Len=2260 TSval=1384331848 TSecr=1384331848
31 33.710749312 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=2308 Win=63872 Len=0 TSval=1384331848 TSecr=1384331848
32 33.710863160 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [FIN, ACK] Seq=2308 Ack=38 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
33 33.712807160 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [FIN, ACK] Seq=38 Ack=2309 Win=64896 Len=0 TSval=1384331850 TSecr=1384331848
34 33.712831919 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=2309 Ack=39 Win=65536 Len=0 TSval=1384331850 TSecr=1384331850
35 39.733846934 127.0.0.1 127.0.0.1 UDP 75 54660 → 6868 Len=33
Frame 30: 2326 bytes on wire (18608 bits), 2326 bytes captured (18608 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 38071, Dst Port: 49510, Seq: 48, Ack: 38, Len: 2260
Data (2260 bytes)
0000 00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00 ..E.
0010 09 08 1c de 40 00 40 06 17 10 7f 00 00 01 7f 00 ....@.@.
0020 00 01 94 b7 c1 66 bc 89 ac fe 12 24 49 85 80 18 .....f. ...$I...
0030 02 00 06 fd 00 00 01 01 08 0a 52 83 3a 48 52 83 ..... R.:HR.
0040 3a 48 23 21 2f 75 73 72 2f 62 69 6e 2f 70 79 74 :H#!/usr /bin/pyt
0050 68 6f 6e 0a 0a 66 72 6f 6d 20 6d 69 6e 69 6e 65 hon fro m minine
0060 74 2e 6e 65 74 20 69 6d 70 6f 72 74 20 4d 69 6e t.net im port Min
0070 69 6e 65 74 0a 66 72 6f 6d 20 6d 69 6e 69 6e 65 inet fro m minine
0080 74 2e 6e 6f 64 65 20 69 6d 70 6f 72 74 20 43 6f t.node i mport Co
0090 6e 74 72 6f 6c 6c 65 72 2c 20 52 65 6d 6f 74 65 ntroller , Remote
00a0 43 6f 6e 74 72 6f 6c 6c 65 72 2c 20 4f 56 53 43 Controll er, OVSC
00b0 6f 6e 74 72 6f 6c 6c 65 72 0a 66 72 6f 6d 20 6d ontrolle r from m
00c0 69 6e 69 6e 65 74 2e 6e 6f 64 65 20 69 6d 70 6f ininet.n ode impo
00d0 72 74 20 43 50 55 4c 69 6d 69 74 65 64 48 6f 73 rt CPULi mitedHos
00e0 74 2c 20 48 6f 73 74 2c 20 4e 6f 64 65 0a 66 72 t, Host, Node fr
00f0 6f 6d 20 6d 69 6e 69 6e 65 74 2e 6e 6f 64 65 20 om minin et.node
0100 69 6d 70 6f 72 74 20 4f 56 53 4b 65 72 6e 65 6c import O VSKernel
0110 53 77 69 74 63 68 2c 20 55 73 65 72 53 77 69 74 Switch, UserSwit
0120 63 68 0a 66 72 6f 6d 20 6d 69 6e 69 6e 65 74 2e ch from mininet.
0130 6e 6f 64 65 20 69 6d 70 6f 72 74 20 49 56 53 53 node imp ort IVSS
0140 77 69 74 63 68 0a 66 72 6f 6d 20 6d 69 6e 69 6e witch fr om minin
0150 65 74 2e 63 6c 69 20 69 6d 70 6f 72 74 20 43 4c et.cli i mport CL
0160 49 0a 66 72 6f 6d 20 6d 69 6e 69 6e 65 74 2e 6c I from m ininet.l
0170 6f 67 20 69 6d 70 6f 72 74 20 73 65 74 4c 6f 67 og impor t setLog
0180 4c 65 76 65 6c 2c 20 69 6e 66 6f 0a 66 72 6f 6d Level, i nfo from
0190 20 6d 69 6e 69 6e 65 74 2e 6c 69 6e 6b 20 69 6d mininet .link im
01a0 70 6f 72 74 20 54 43 4c 69 6e 6b 2c 20 49 6e 74 port TCL ink, Int
01b0 66 0a 66 72 6f 6d 20 73 79 73 2e 70 72 6f 63 65 f.from s ubproces
wiresharkDescargaFicheros.pcapng
Paquetes: 49 · Mostrado: 49 (100.0%)
Perfil: Default


wiresharkDescargaFicheros.pcapng
Archivo Edición Visualización Ir Captura Analizar Estadísticas Telefonía Wireless Herramientas Ayuda
Aplique un filtro de visualización ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
19 33.703097045 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=13 Win=65536 Len=0 TSval=1384331840 TSecr=1384331840
20 33.703159239 127.0.0.1 127.0.0.1 TCP 73 49510 → 38071 [PSH, ACK] Seq=13 Ack=1 Win=65536 Len=7 TSval=1384331840 TSecr=1384331840
21 33.703164899 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=20 Win=65536 Len=0 TSval=1384331840 TSecr=1384331840
22 33.703311828 127.0.0.1 127.0.0.1 TCP 84 49510 → 38071 [PSH, ACK] Seq=20 Ack=1 Win=65536 Len=18 TSval=1384331841 TSecr=1384331840
23 33.703318312 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=1 Ack=38 Win=65536 Len=0 TSval=1384331841 TSecr=1384331841
24 33.710368545 127.0.0.1 127.0.0.1 TCP 67 38071 → 49510 [PSH, ACK] Seq=1 Ack=38 Win=65536 Len=1 TSval=1384331848 TSecr=1384331841
25 33.710382815 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=2 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
26 33.710600624 127.0.0.1 127.0.0.1 TCP 108 38071 → 49510 [PSH, ACK] Seq=2 Ack=38 Win=65536 Len=42 TSval=1384331848 TSecr=1384331848
27 33.710607474 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=44 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
28 33.710688580 127.0.0.1 127.0.0.1 TCP 70 38071 → 49510 [PSH, ACK] Seq=44 Ack=38 Win=65536 Len=4 TSval=1384331848 TSecr=1384331848
29 33.710692597 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=48 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
30 33.710744446 127.0.0.1 127.0.0.1 TCP 2326 38071 → 49510 [PSH, ACK] Seq=48 Ack=38 Win=65536 Len=2260 TSval=1384331848 TSecr=1384331848
31 33.710749312 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [ACK] Seq=38 Ack=2308 Win=63872 Len=0 TSval=1384331848 TSecr=1384331848
32 33.710863160 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [FIN, ACK] Seq=2308 Ack=38 Win=65536 Len=0 TSval=1384331848 TSecr=1384331848
33 33.712807169 127.0.0.1 127.0.0.1 TCP 66 49510 → 38071 [FIN, ACK] Seq=38 Ack=2309 Win=64896 Len=0 TSval=1384331850 TSecr=1384331848
34 33.712831919 127.0.0.1 127.0.0.1 TCP 66 38071 → 49510 [ACK] Seq=2309 Ack=39 Win=65536 Len=0 TSval=1384331850 TSecr=1384331850
35 39.733846934 127.0.0.1 127.0.0.1 UDP 75 54660 → 6868 Len=33
36 39.734311117 127.0.0.1 127.0.0.1 UDP 94 6868 → 54660 Len=52
37 40.706609548 127.0.0.1 127.0.0.1 TCP 66 49074 → 46811 [FIN, ACK] Seq=1 Ack=1 Win=512 Len=0 TSval=1384338844 TSecr=1384289292
38 40.707921135 127.0.0.1 127.0.0.1 TCP 66 46811 → 49074 [FIN, ACK] Seq=1 Ack=2 Win=512 Len=0 TSval=1384338845 TSecr=1384338844
39 40.707961801 127.0.0.1 127.0.0.1 TCP 66 49074 → 46811 [ACK] Seq=2 Ack=2 Win=512 Len=0 TSval=1384338845 TSecr=1384338845
40 44.721764420 127.0.0.1 127.0.0.1 UDP 85 53235 → 6868 Len=43
41 44.722717504 127.0.0.1 127.0.0.1 UDP 72 6868 → 53235 Len=30
42 46.724099857 127.0.0.1 127.0.0.1 UDP 74 53235 → 6868 Len=32
43 46.725125495 127.0.0.1 127.0.0.1 UDP 92 6868 → 53235 Len=50
44 47.802422683 127.0.0.1 127.0.0.1 TCP 66 59698 → 36007 [FIN, ACK] Seq=1 Ack=1 Win=512 Len=0 TSval=1384345940 TSecr=1384287752
Frame 40: 85 bytes on wire (680 bits), 85 bytes captured (680 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 53235, Dst Port: 6868
Data (43 bytes)
0000 00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00
0010 00 47 58 0f 40 00 40 11 e4 94 7f 00 00 01 7f 00
0020 00 01 cf f3 1a d4 00 33 fe 46 6f 70 65 72 61 74
0030 69 6f 6e 3a 75 6e 72 65 67 69 73 74 65 72 5f 73
0040 65 72 76 65 72 0a 73 65 73 73 69 6f 6e 6b 65 79
0050 3a 32 35 0a 0a
wiresharkDescargaFicheros.pcapng
Paquetes: 49 · Mostrado: 49 (100.0%)
Perfil: Default


wiresharkNanofilesDirectorio.pcapng
Archivo Edición Visualización Ir Captura Analizar Estadísticas Telefonía Wireless Herramientas Ayuda
Aplique un filtro de visualización ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
3 5.853691066 127.0.0.1 127.0.0.1 UDP 75 60946 → 6868 Len=33
4 5.854079916 127.0.0.1 127.0.0.1 UDP 66 6868 → 60946 Len=24
5 12.932666734 127.0.0.1 127.0.0.1 UDP 76 44595 → 6868 Len=34
6 12.933747554 127.0.0.1 127.0.0.1 UDP 93 6868 → 44595 Len=51
7 16.664354712 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
8 16.675950017 127.0.0.1 127.0.0.1 UDP 128 6868 → 44595 Len=86
9 28.414820564 127.0.0.1 127.0.0.1 UDP 95 51482 → 6868 Len=53
10 28.415869317 127.0.0.1 127.0.0.1 UDP 70 6868 → 51482 Len=28
11 28.431336377 127.0.0.1 127.0.0.1 UDP 146 51482 → 6868 Len=104
12 28.431813372 127.0.0.1 127.0.0.1 UDP 63 6868 → 51482 Len=21
13 33.753848672 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
14 33.755632269 127.0.0.1 127.0.0.1 UDP 143 6868 → 44595 Len=101
15 37.067734208 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
16 37.069125621 127.0.0.1 127.0.0.1 UDP 141 6868 → 44595 Len=99
17 45.454178755 127.0.0.1 127.0.0.1 UDP 86 51482 → 6868 Len=44
18 45.454648885 127.0.0.1 127.0.0.1 UDP 72 6868 → 51482 Len=30
19 51.108312629 127.0.0.1 127.0.0.1 UDP 95 51482 → 6868 Len=53
20 51.109128935 127.0.0.1 127.0.0.1 UDP 70 6868 → 51482 Len=28
21 51.109732857 127.0.0.1 127.0.0.1 UDP 146 51482 → 6868 Len=104
22 51.110712172 127.0.0.1 127.0.0.1 UDP 63 6868 → 51482 Len=21
23 57.625518705 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
24 57.626897861 127.0.0.1 127.0.0.1 UDP 143 6868 → 44595 Len=101
25 67.043257519 127.0.0.1 127.0.0.1 UDP 121 44595 → 6868 Len=79
26 67.044062128 127.0.0.1 127.0.0.1 UDP 78 6868 → 44595 Len=36
27 69.988478589 127.0.0.1 127.0.0.1 UDP 95 44595 → 6868 Len=53
28 69.989454585 127.0.0.1 127.0.0.1 UDP 70 6868 → 44595 Len=28
Frame 8: 128 bytes on wire (1024 bits), 128 bytes captured (1024 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 6868, Dst Port: 44595
Data (86 bytes)
0000 00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00
0010 00 72 10 ae 40 00 40 11 2b cb 7f 00 00 01 7f 00
0020 00 01 1a d4 ae 33 00 5e fe 71 6f 70 65 72 61 74
0030 69 6f 6e 3a 75 73 65 72 6c 69 73 74 6f 6b 0a 6e
0040 69 63 6b 6e 61 6d 65 3a 63 6c 69 65 6e 74 65 0a
0050 73 65 73 73 69 6f 6e 6b 65 79 3a 34 31 30 0a 75
0060 73 65 72 73 3a 73 65 72 76 65 72 3a 55 73 65 72
0070 2c 63 6c 69 65 6e 74 65 3a 55 73 65 72 2c 0a 0a
wiresharkNanofilesDirectorio.pcapng
Paquetes: 49 · Mostrado: 49 (100.0%)
Perfil: Default


wiresharkNanofilesDirectorio.pcapng
Archivo Edición Visualización Ir Captura Analizar Estadísticas Telefonía Wireless Herramientas Ayuda
Aplique un filtro de visualización ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
3 5.853691066 127.0.0.1 127.0.0.1 UDP 75 60946 → 6868 Len=33
4 5.854079916 127.0.0.1 127.0.0.1 UDP 66 6868 → 60946 Len=24
5 12.932666734 127.0.0.1 127.0.0.1 UDP 76 44595 → 6868 Len=34
6 12.933747554 127.0.0.1 127.0.0.1 UDP 93 6868 → 44595 Len=51
7 16.664354712 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
8 16.675990837 127.0.0.1 127.0.0.1 UDP 128 6868 → 44595 Len=86
9 28.414820564 127.0.0.1 127.0.0.1 UDP 95 51482 → 6868 Len=53
10 28.415869317 127.0.0.1 127.0.0.1 UDP 70 6868 → 51482 Len=28
11 28.431336377 127.0.0.1 127.0.0.1 UDP 146 51482 → 6868 Len=104
12 28.431813372 127.0.0.1 127.0.0.1 UDP 63 6868 → 51482 Len=21
13 33.753848672 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
14 33.755632269 127.0.0.1 127.0.0.1 UDP 143 6868 → 44595 Len=101
15 37.067734208 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
16 37.069125621 127.0.0.1 127.0.0.1 UDP 141 6868 → 44595 Len=99
17 45.454178755 127.0.0.1 127.0.0.1 UDP 86 51482 → 6868 Len=44
18 45.454648885 127.0.0.1 127.0.0.1 UDP 72 6868 → 51482 Len=30
19 51.108312629 127.0.0.1 127.0.0.1 UDP 95 51482 → 6868 Len=53
20 51.109128935 127.0.0.1 127.0.0.1 UDP 70 6868 → 51482 Len=28
21 51.109732857 127.0.0.1 127.0.0.1 UDP 146 51482 → 6868 Len=104
22 51.110712172 127.0.0.1 127.0.0.1 UDP 63 6868 → 51482 Len=21
23 57.625518705 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
24 57.626897861 127.0.0.1 127.0.0.1 UDP 143 6868 → 44595 Len=101
25 67.043257519 127.0.0.1 127.0.0.1 UDP 121 44595 → 6868 Len=79
26 67.044062128 127.0.0.1 127.0.0.1 UDP 78 6868 → 44595 Len=36
27 69.980478589 127.0.0.1 127.0.0.1 UDP 95 44595 → 6868 Len=53
28 69.980454585 127.0.0.1 127.0.0.1 UDP 70 6868 → 44595 Len=28
Frame 21: 146 bytes on wire (1168 bits), 146 bytes captured (1168 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 51482, Dst Port: 6868
Data (104 bytes)
0000 00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00 .... E.
0010 00 84 2c f0 40 00 40 11 0f 77 7f 00 00 01 7f 00 ..,.@.@. .w.....
0020 00 01 c9 1a 1a d4 00 70 fe 83 6f 70 65 72 61 74 ..... p..operat
0030 69 6f 6e 3a 70 75 62 6c 69 73 68 0a 73 65 73 73 ion:publ ish sess
0040 69 6f 6e 6b 65 79 3a 39 30 38 0a 66 69 6c 65 73 ionkey:9 08 files
0050 3a 62 32 31 63 63 33 34 38 61 31 35 61 35 31 62 :b21cc34 8a15a51b
0060 36 30 34 33 32 61 62 34 38 39 31 61 38 32 36 36 60432ab4 891a8266
0070 61 65 30 32 30 66 63 61 61 2c 74 6f 70 6f 6c 6f ae020fca a,topolo
0080 67 69 61 5f 65 6a 65 2e 70 79 2c 32 32 36 30 3a gia_eje. py,2260:
0090 0a 0a
wiresharkNanofilesDirectorio.pcapng
Paquetes: 49 · Mostrado: 49 (100.0%)
Perfil: Default


wiresharkNanofilesDirectorio.pcapng
Archivo Edición Visualización Ir Captura Analizar Estadísticas Telefonía Wireless Herramientas Ayuda
Aplique un filtro de visualización ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
15 37.067734208 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
16 37.069125621 127.0.0.1 127.0.0.1 UDP 141 6868 → 44595 Len=99
17 45.454178755 127.0.0.1 127.0.0.1 UDP 86 51482 → 6868 Len=44
18 45.454648885 127.0.0.1 127.0.0.1 UDP 72 6868 → 51482 Len=30
19 51.108312629 127.0.0.1 127.0.0.1 UDP 95 51482 → 6868 Len=53
20 51.109128935 127.0.0.1 127.0.0.1 UDP 70 6868 → 51482 Len=28
21 51.109732857 127.0.0.1 127.0.0.1 UDP 146 51482 → 6868 Len=104
22 51.110712172 127.0.0.1 127.0.0.1 UDP 63 6868 → 51482 Len=21
23 57.625518705 127.0.0.1 127.0.0.1 UDP 77 44595 → 6868 Len=35
24 57.626897861 127.0.0.1 127.0.0.1 UDP 143 6868 → 44595 Len=101
25 67.043257519 127.0.0.1 127.0.0.1 UDP 121 44595 → 6868 Len=79
26 67.044062128 127.0.0.1 127.0.0.1 UDP 78 6868 → 44595 Len=36
27 69.980478589 127.0.0.1 127.0.0.1 UDP 95 44595 → 6868 Len=53
28 69.980454585 127.0.0.1 127.0.0.1 UDP 70 6868 → 44595 Len=28
29 69.994658872 127.0.0.1 127.0.0.1 UDP 146 44595 → 6868 Len=104
30 69.994916195 127.0.0.1 127.0.0.1 UDP 63 6868 → 44595 Len=21
31 73.706896367 127.0.0.1 127.0.0.1 UDP 121 44595 → 6868 Len=79
32 73.707101534 127.0.0.1 127.0.0.1 UDP 86 6868 → 44595 Len=44
33 79.006190869 127.0.0.1 127.0.0.1 UDP 86 44595 → 6868 Len=44
34 79.007313980 127.0.0.1 127.0.0.1 UDP 72 6868 → 44595 Len=30
35 81.825457530 127.0.0.1 127.0.0.1 UDP 75 44595 → 6868 Len=33
36 81.826551663 127.0.0.1 127.0.0.1 UDP 94 6868 → 44595 Len=52
37 83.685027701 127.0.0.1 127.0.0.1 TCP 66 60856 → 36987 [FIN, ACK] Seq=1 Ack=1 Win=512 Len=0 TSval=1383673368 TSecr=1383565719
38 83.686327779 127.0.0.1 127.0.0.1 TCP 66 36987 → 60856 [FIN, ACK] Seq=1 Ack=2 Win=512 Len=0 TSval=1383673369 TSecr=1383673368
39 83.686353306 127.0.0.1 127.0.0.1 TCP 66 60856 → 36987 [ACK] Seq=2 Ack=2 Win=512 Len=0 TSval=1383673369 TSecr=1383673369
40 87.155581847 127.0.0.1 127.0.0.1 UDP 86 51482 → 6868 Len=44
Frame 29: 146 bytes on wire (1168 bits), 146 bytes captured (1168 bits) on interface lo, id 0
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 44595, Dst Port: 6868
Data (104 bytes)
0000 00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00 .... E.
0010 00 84 33 b2 40 00 40 11 08 b5 7f 00 00 01 7f 00 ..3.@.@. .....
0020 00 01 ae 33 1a d4 00 70 fe 83 6f 70 65 72 61 74 ...3...p ..operat
0030 69 6f 6e 3a 70 75 62 6c 69 73 68 0a 73 65 73 73 ion:publ ish sess
0040 69 6f 6e 6b 65 79 3a 34 31 30 0a 66 69 6c 65 73 ionkey:4 10 files
0050 3a 62 32 31 63 63 33 34 38 61 31 35 61 35 31 62 :b21cc34 8a15a51b
0060 36 30 34 33 32 61 62 34 38 39 31 61 38 32 36 36 60432ab4 891a8266
0070 61 65 30 32 30 66 63 61 61 2c 74 6f 70 6f 6c 6f ae020fca a,topolo
0080 67 69 61 5f 65 6a 65 2e 70 79 2c 32 32 36 30 3a gia_eje. py,2260:
0090 0a 0a
wiresharkNanofilesDirectorio.pcapng
Paquetes: 49 · Mostrado: 49 (100.0%)
Perfil: Default