Bienvenidos a la clase 65 de **HDC** :),

¿Están listos? Proactividad.

“¡Eso! ¡Procrastividad!”

Eh... Bueno, está bien. Hoy veremos **FTP**. Hasta ahora lo que vimos fue...

“Robar cosas con malware.”

… ¿De verdad? ¿Éso has aprendido? El tema es más extenso que “robar”. **FTP** o **File Transfer Protocol**, es decir una manera de **transferir archivos**.

“¿De la misma manera que lo hago por Google Drive?”

Buena pregunta. Google Drive es otro tipo de servicio que nada tiene que ver con el FTP. Veamos ciertas cuestiones.

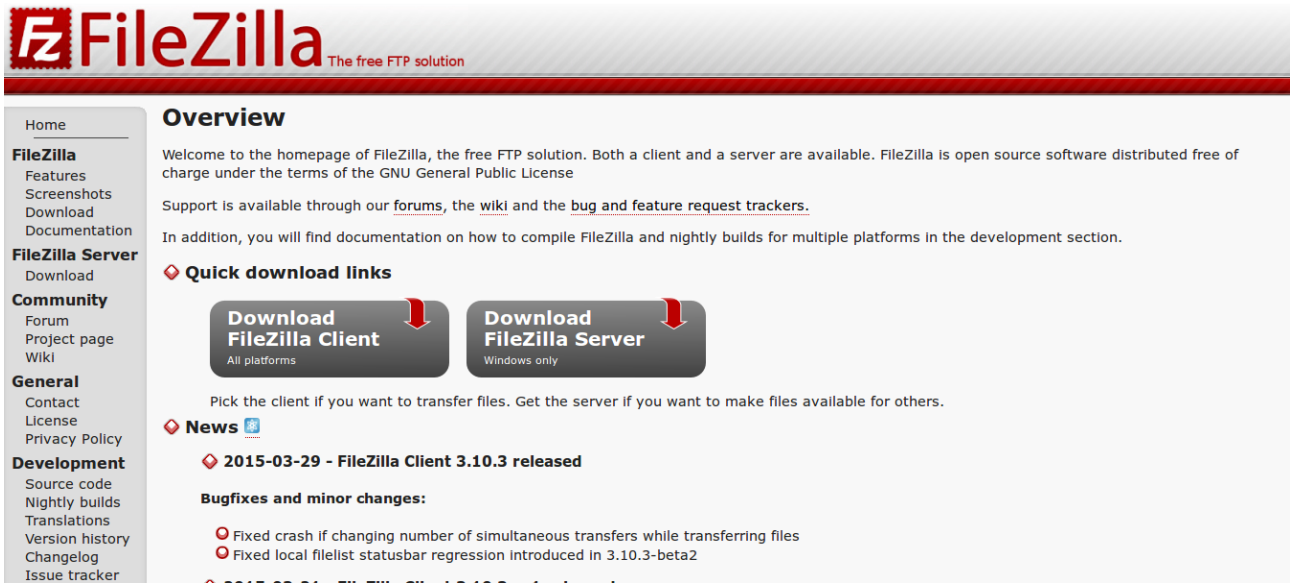
La idea principal es el **servidor**. Un servidor FTP instalado en algun equipo que **alojará archivos** y los **clientes** pueden conectarse a él y descargarse archivos o subirlos (todo depende de los **permisos** que se le brinden al usuario). Obviamente trabaja mediante **TCP** porque no es necesario que sea rápido pero sí efectivo.

Para hacer una clase interesante, vamos a instalar en nuestra máquina virtual con W7, un server FTP.

“Quiero hackear.”

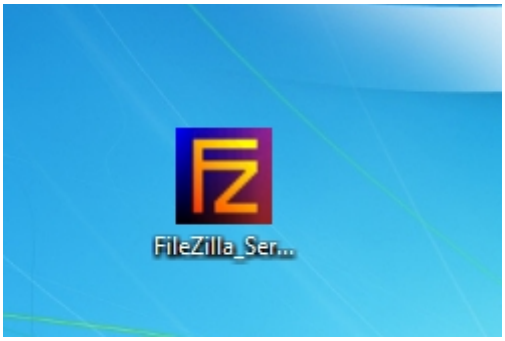
Manolo, primero caminar y luego correr. Sé que el camino es duro y largo pero te aseguro que es satisfactorio :).

Vamos a descargar **FileZilla** desde la página oficial (es **gratuito**, quédense tranquilos): https://filezilla-project.org/

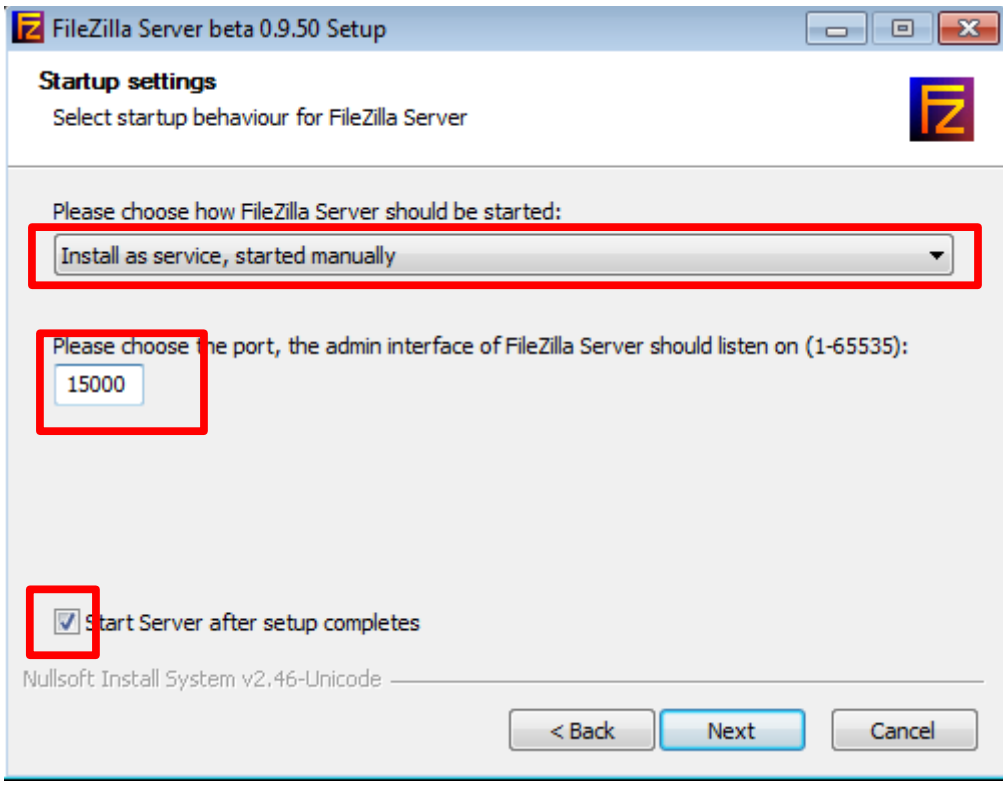


El **cliente** lo descargan para su máquina **host**, y el **Filezilla Server** lo descargan en la máquina **virtual**. Aunque si quieren, pueden instalarlo al **revés** o los dos en el mismo equipo -que es lo que haré yo-.

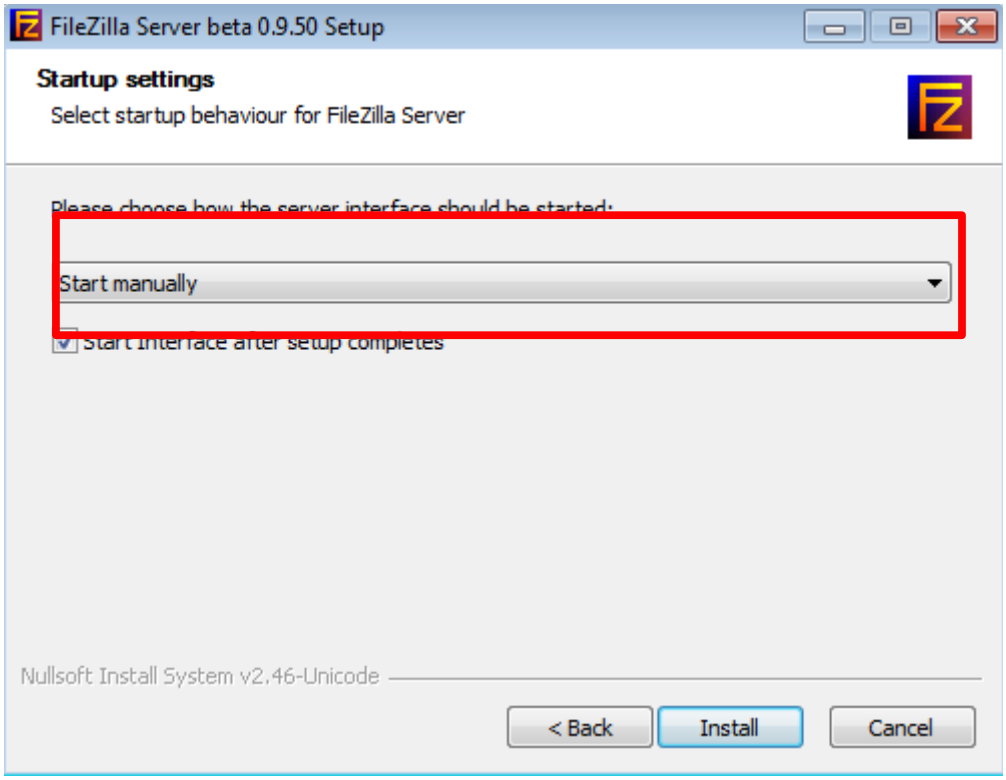
Bueno, ya descargado luego de unos segundos, le damos a instalar al **exe** del server.



Le vamos a dar "***next***" hasta que aparezca esta ventana a la que vamos a dejar que Filezilla se instale como un **servicio** pero que **no empiece automáticamente** sino que nosotros le vamos a dar la orden. El puerto puede ser cualquiera pero yo prefiero elegir algo que recuerde (aunque es indistinto, no eligan los puertos que son estándar para otros servicios) y voy a querer que inicie el server una vez terminada la instalación, sobre todo para saber si anda correctamente.



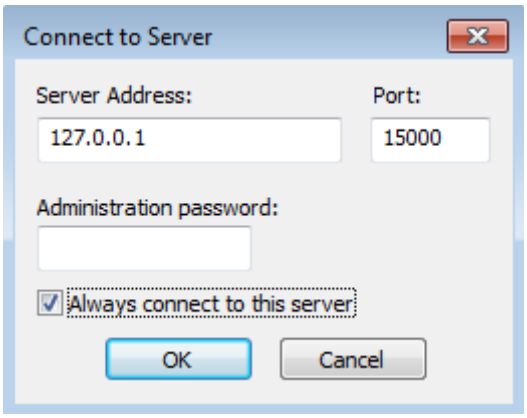
Luego de darle "***next***" nuevamente, aparecerá esta otra ventana.



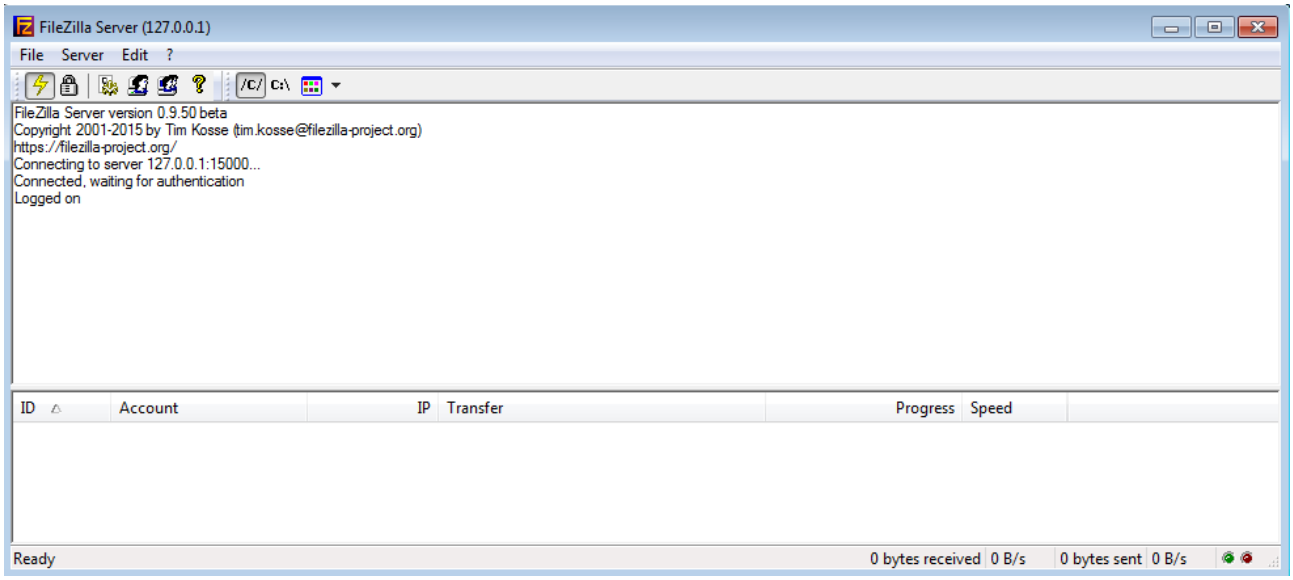
Una vez terminada la instalación del server, aparecerá esta ventanita la cual aparece porque iniciamos el Filezilla.

En donde dice "**Server Address**", apuntaremos a **nuestra IP** (recuerden que el servicio del server lo instalamos en nuestro equipo pero Filezilla puede actuar solo como interfaz gráfica de un

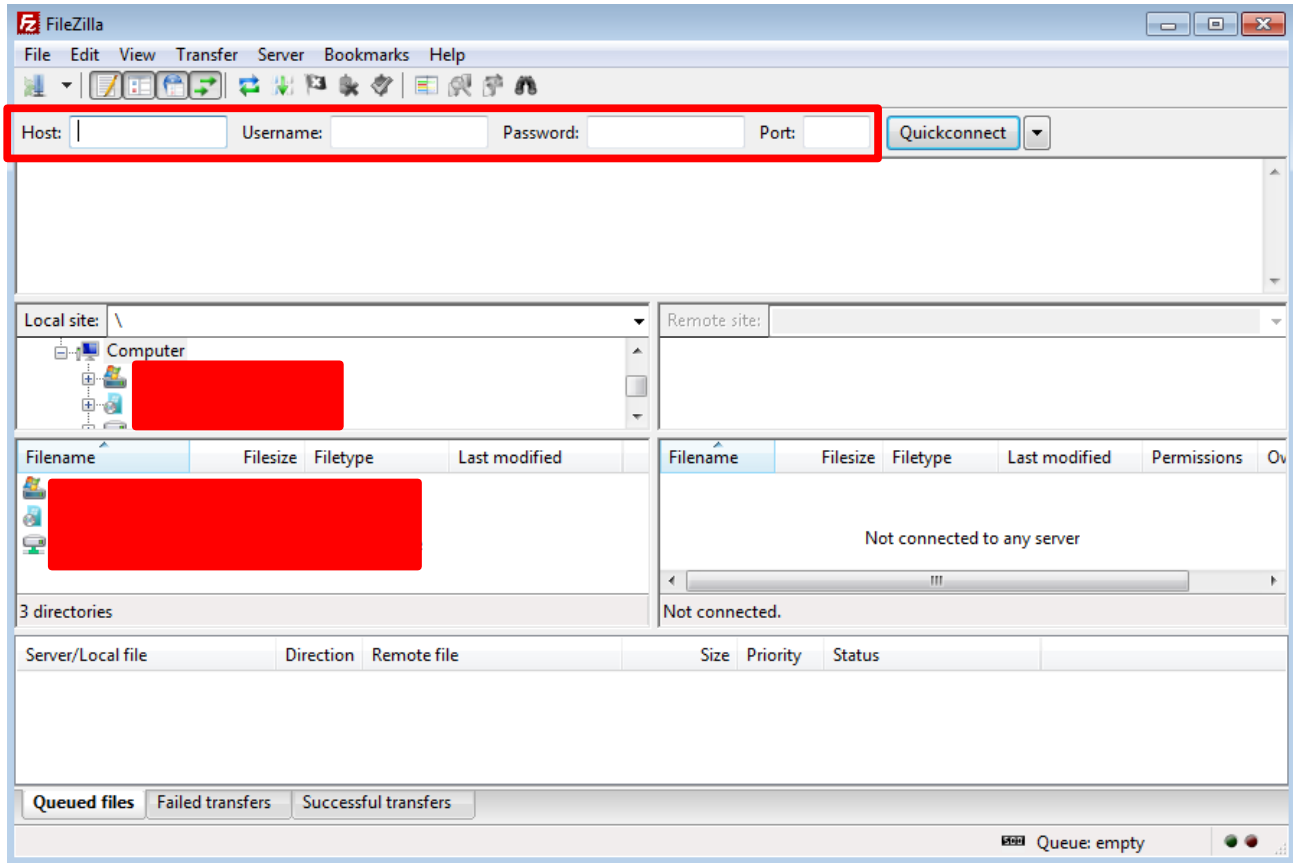
servidor FTP externo) que como yo estoy conectado a un **DHCP** (¿ésto lo explique anteriormente?), mi IP **no es fija** en la red interna. Lo mejor que puedo hacer es apuntar a la IP definida como un **sinónimo** de **LocalHost**. En la password dejen el cuadro en blanco.



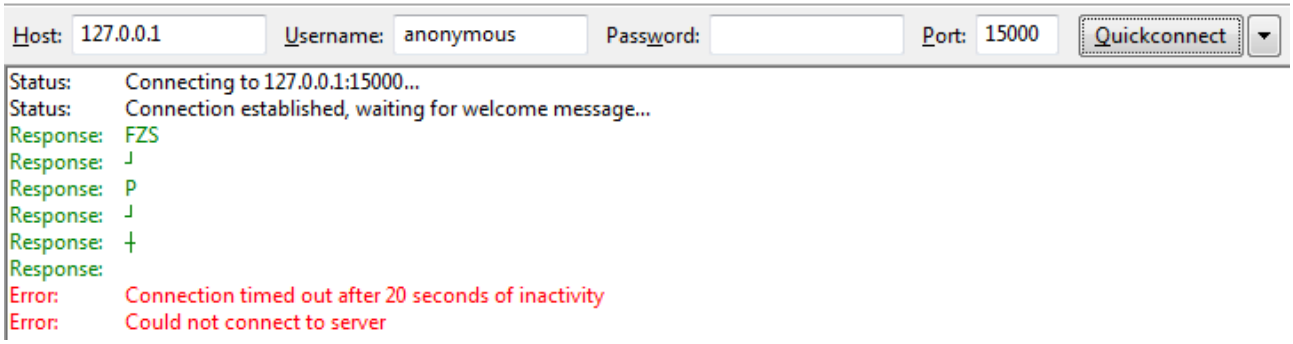
Entonces aparecerá la interfaz y si todo sale bien, como última linea el "***Logged on***" de su correcto logueo.



Ahora a la otra parte. Pero tengan **cuidado** al instalar el **cliente**, que intenta de instalarte unos adware desde el descargador de **SourceForge** (tuve que darle al Decline un par de veces). Va sin imágenes porque no hay opciones que tocar. Cuando termina nos aparece la interfaz del cliente.



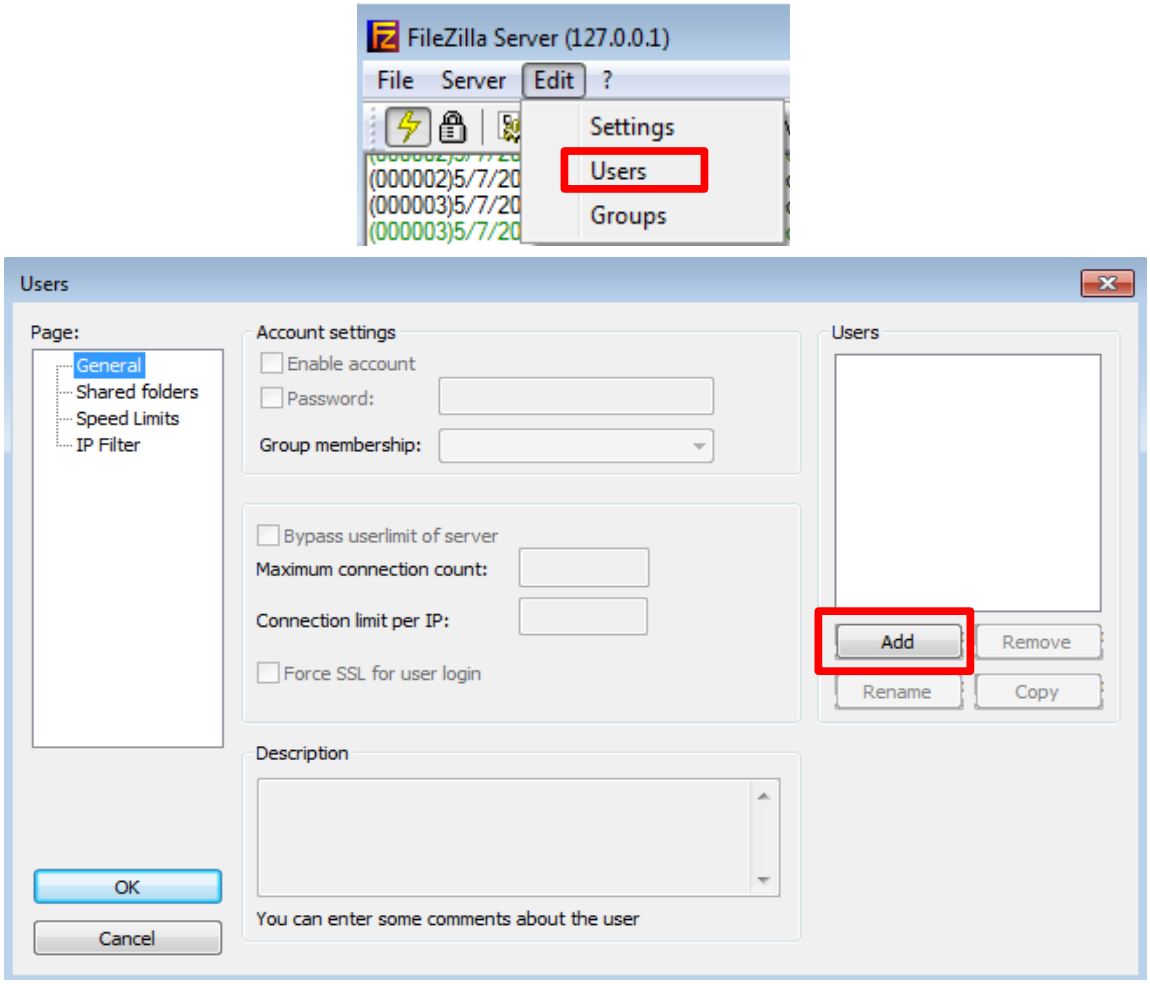
En donde está el recuadro rojo vamos a poner los datos que sabemos. El host es la dirección IP de donde esté el server. En mi caso, es mi **localhost** por lo que la **IP** es **127.0.0.1**. El **puerto** es el **15000** y el usuario y password estarán en blanco. Una vez le dimos a Quickconnect que aparece al lado, se conectará. Pero vemos que pasa algo más luego de 20 segundos.



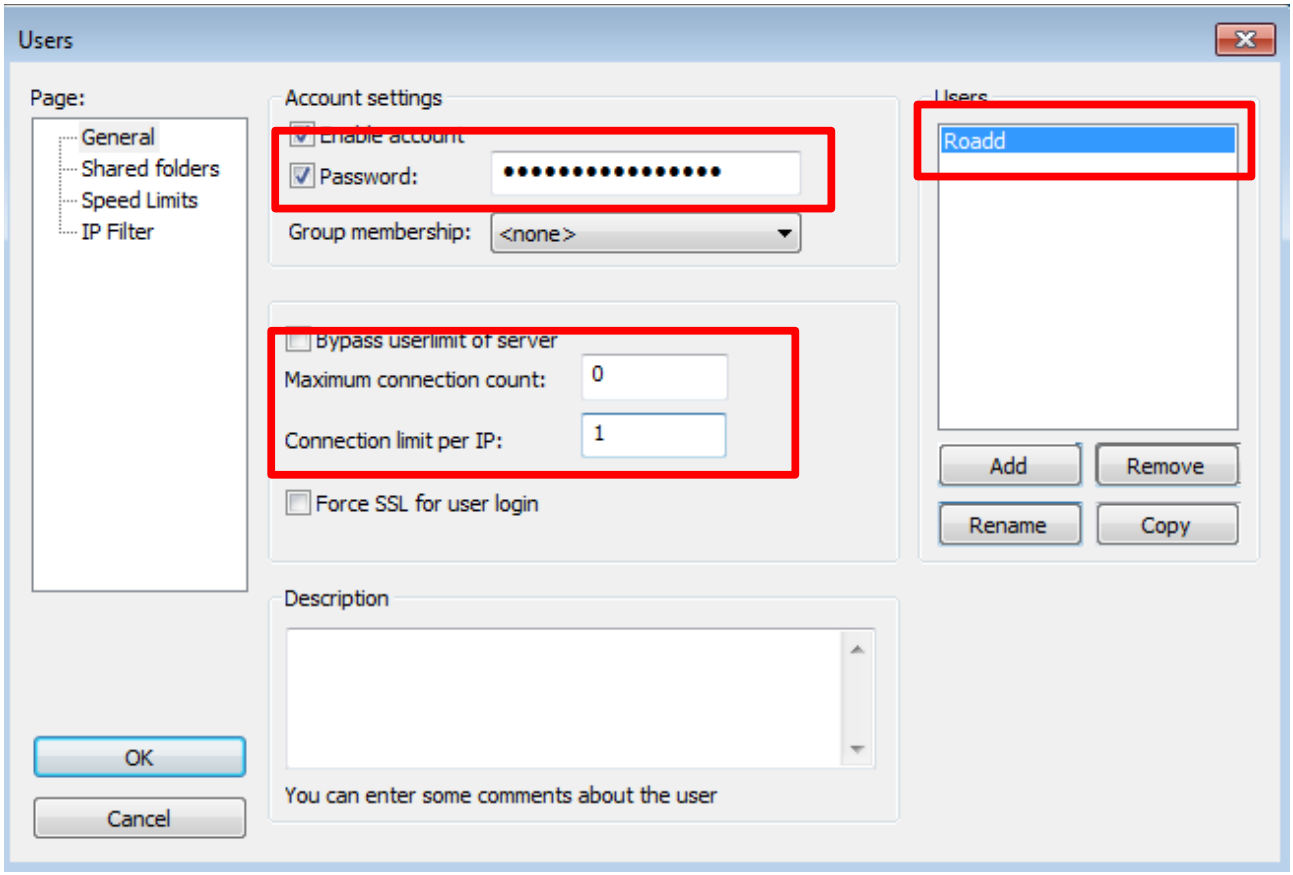
Las primeras dos líneas son el estado de conexión establecido. Pero luego aparecen dos errores. Dice que **si por 20 segundos no tuviste actividad, la conexión caerá**.

“¿Veinte segundos? Eso es muy poco, con razón los hackers hacen todo rápido.”

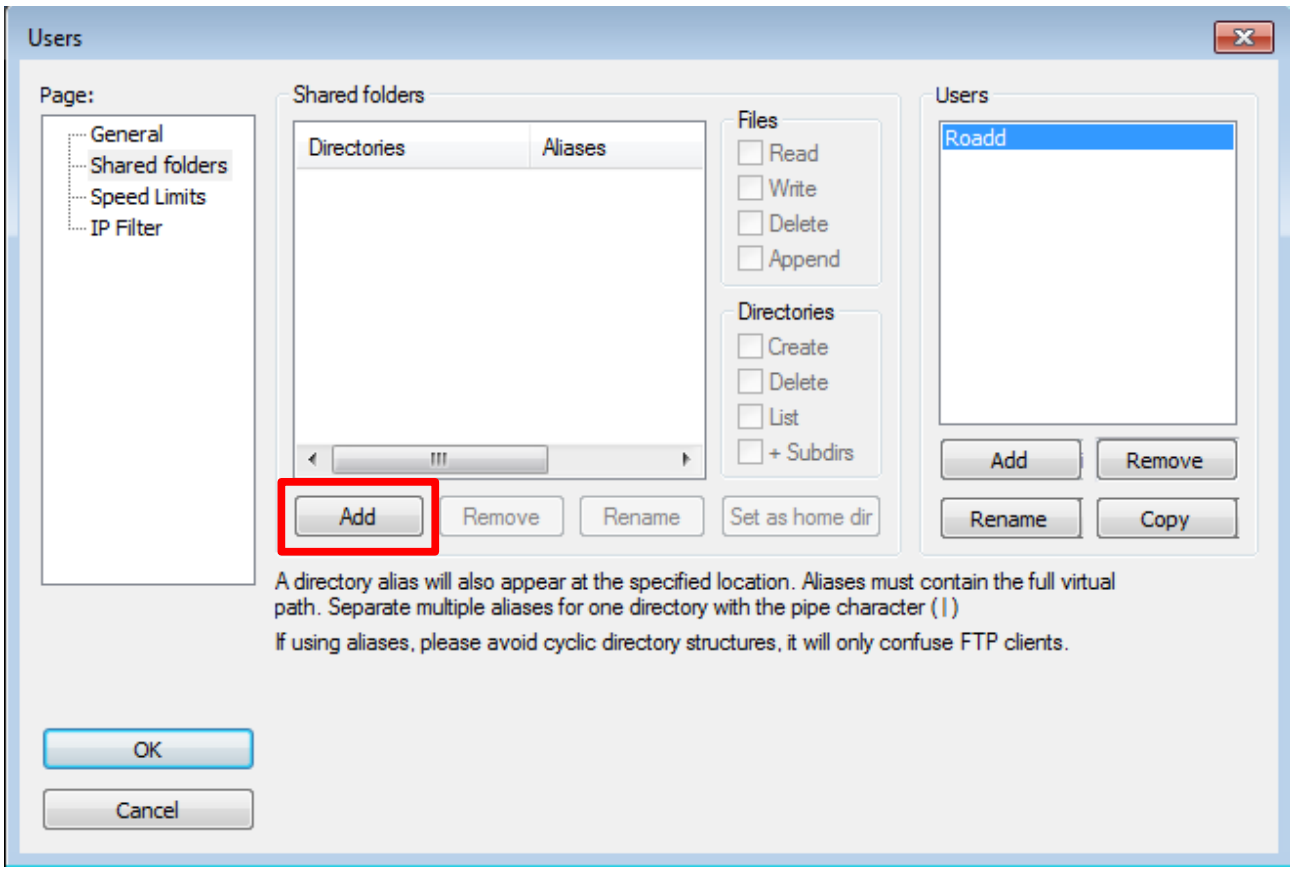
Veinte segundos es **mucho tiempo** porque cada vez que hagas algo esos 20 segundos se renuevan. Y no tiene nada que ver con lo que estás pensando. Hagamos una práctica para subir y descargar archivos. Para ésto, primero vamos a **añadir un usuario** en el servidor FTP. Vamos a ***Edit → Users***. Y allí dentro, en ***Add***, podemos crearlo.



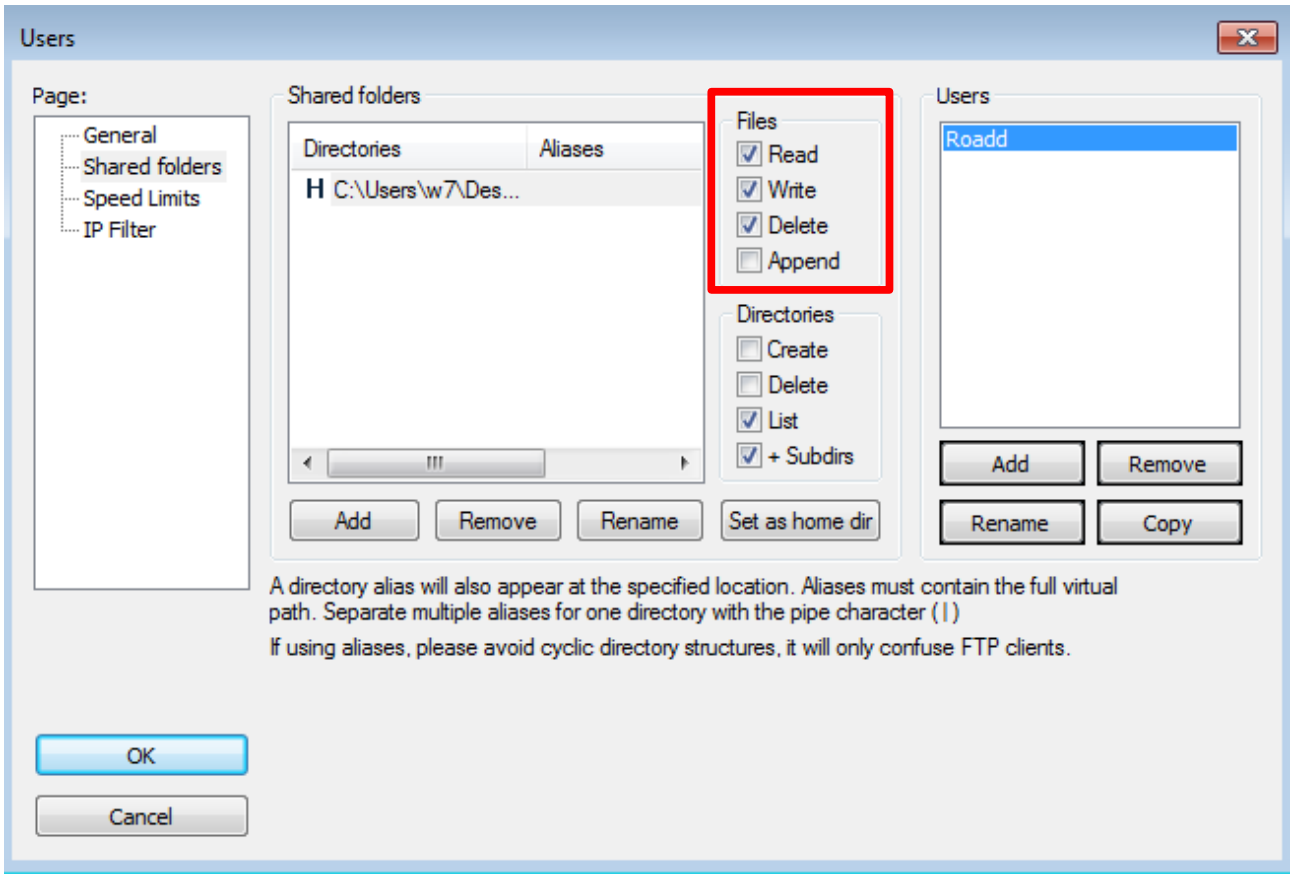
Aquí tenemos una ventana donde podemos **agregar el nombre del usuario** -en mi caso ***Roadd***- y luego **tildamos** la opción de **password** con una que queramos nosotros. Vean que también aparece la opcion de agregar ese usuario a un **grupo** determinado, para que podamos administrar permisos de a muchos pero **en este caso no es necesario**. Lo que aparece sobre ***Maximum connection count*** es la **cantidad máxima de conexiones** con este mismo usuario; y ***Connection limit per IP*** es la cantidad de conexiones **limite por IP** (redundante, si lo leen es obvio). Cuando los valores están en cero corresponden a un "**sin límite**".



No podemos irnos sin agregarle un directorio por el cual va a operar el usuario, en la izquierda tenemos la opcion de ***Shared folders***. Allí, en la parte izquierda, podemos ver que manejamos los directorios. Añadan uno que quieran, yo creé una carpeta en el escritorio y agregaré ésa.

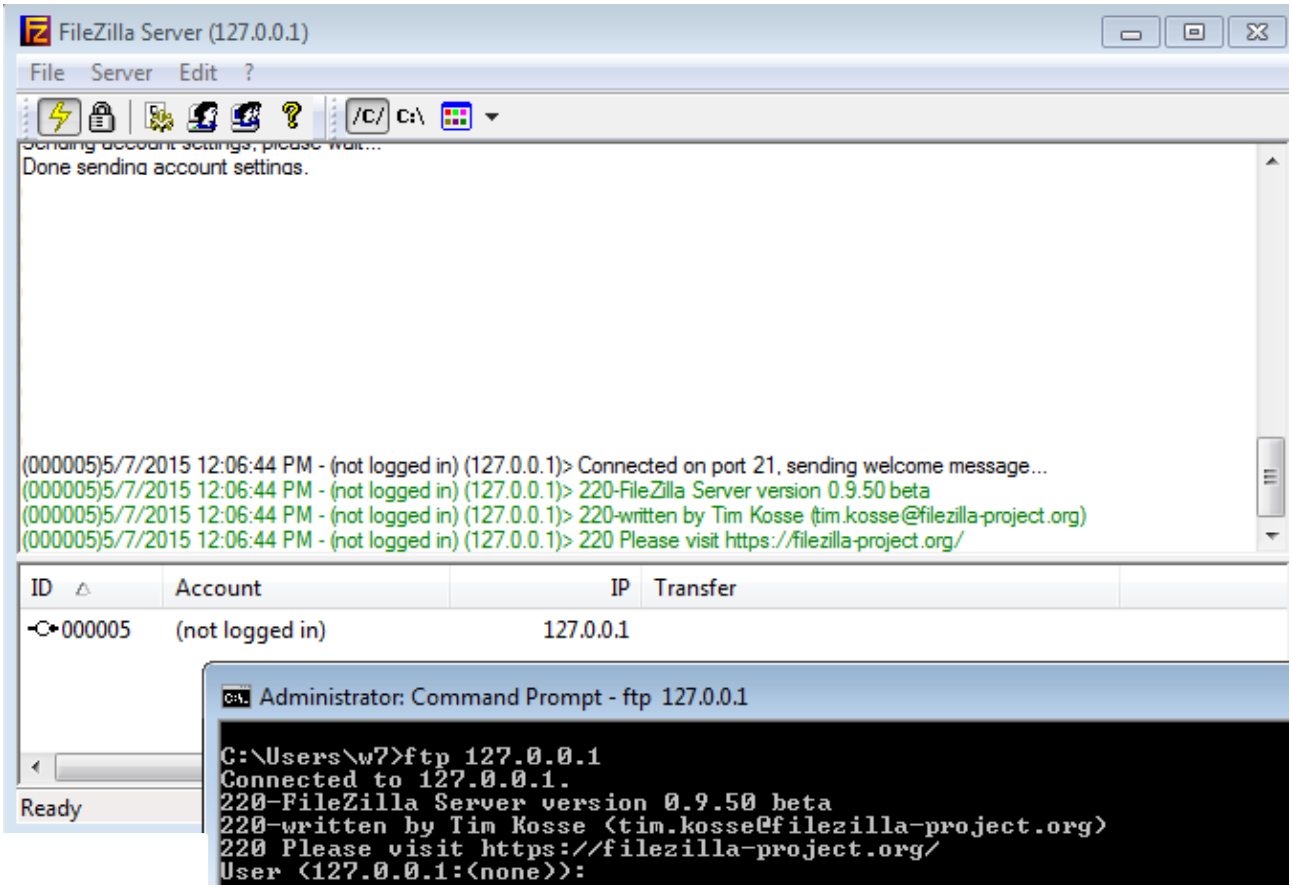


Cuando la agregamos, vemos que los botones que estaban en gris, se habilitan. Voy a habilitar las primeras tres opciones del ***checkbox*** "**Files**". Ya sabrán a qué se refieren :). Y luego ya aceptamos todo.



Abramos una **línea de comandos**. Sé que les hice instalar un cliente del Filezilla y quisiera que lo **investiguen** por su cuenta, y mientras tanto yo les enseño a usar las herramientas que son más **flexibles** ~~aunque un poco feas.~~

En fin, para abrir la conexión tenemos el comando "**ftp <ip>**". Recuerden que mi IP es mi propio localhost, por lo que será "**ftp 127.0.0.1**"



"¡Muchos mensajes! Ésto ya es muy complicado. Me voy."

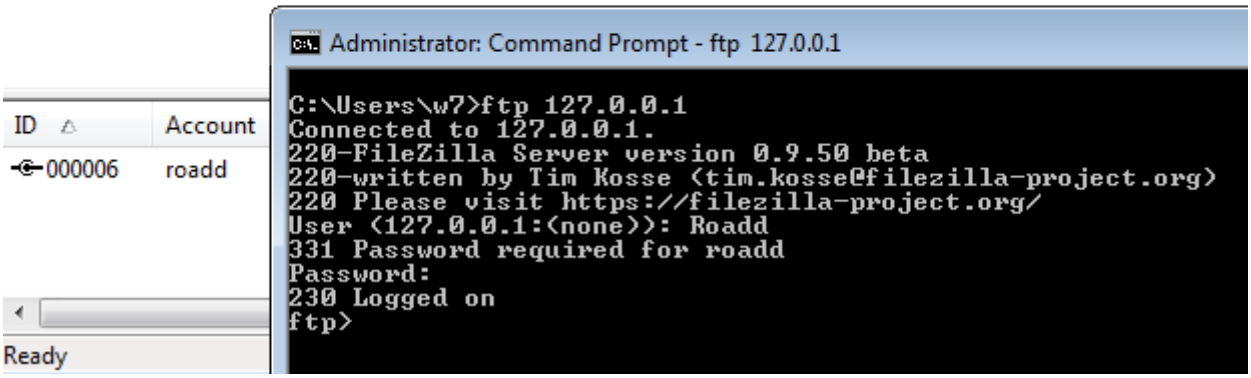
¡Manolo! ¡Volvé! Vas a ver que todo es **informativo** y que no solamente es **necesario**, sino que es muy **interesante**. Lo primero que vemos de interesante es dentro del Filezilla Server, donde aparecen **4 mensajes**. Los mensajes tienen **separaciones**, veamos de qué se tratan.

- **(000005)**: Esto corresponde al ID de la conexión, porque en un server aparecen varios usuarios, varias conexiones y así tenemos una referencia de quién hablamos. Fíjense que también tenemos información de esa ID en la separación de abajo.
- **5/7/2015**: Fecha en formato mm/dd/aaaa.
- **12:06:44 PM**: Hora en formato hh:mm:ss.
- **(not logged in)**: Indica que no está logueado con ningún usuario desde esa IP.
- **(127.0.0.1)**: Cuál es la IP de donde se está haciendo la conexión.
- >: Luego de este símbolo aparece el mensaje o los estados del server.

La primer línea que aparece en negro es información sobre el puerto donde se conecta; lo que aparece en **verde** es el mensaje que se **envía y recibimos** desde el **cliente**. Allí, en nuestro **cmd**, tenemos la **IP** a donde nos conectamos (la mía es la del localhost), y el mensaje que me envío el

server. Para terminar me indica que me pide un **usuario** para poder conectarme y luego una **contraseña**.

```
(000006)5/7/2015 13:21:19 PM - (not logged in) (127.0.0.1)> 220 Please visit https://filezilla-project.org/
(000006)5/7/2015 13:21:24 PM - (not logged in) (127.0.0.1)> USER Roadd
(000006)5/7/2015 13:21:24 PM - (not logged in) (127.0.0.1)> 331 Password required for roadd
(000006)5/7/2015 13:21:31 PM - (not logged in) (127.0.0.1)> PASS **********
(000006)5/7/2015 13:21:31 PM - roadd (127.0.0.1)> 230 Logged on
```



Allí tenemos la manera que me conecté. **Cuando ponen su password,** mientras escriben no van a ver asteriscos ni nada, simplemente **nada se mueve**. Ésto aparece para que nadie sepa **cuántas** letras -o mejor dicho caracteres- escribiste. Una vez terminado nos da el **prompt** para que demos comandos.

Hay un par de cosas a las cuales prestar atención:

- **El ID de la conexión cambió**: esto sucedió porque se me cerro la conexión anterior por lo que tuve que abrir otra (acostúmbrense porque va a pasarme seguido).
- **Hay unos numeritos antes de los mensajes del server**: esto corresponde a los ID de los mensajes.

Con ***help*** o ***?*** listamos los comandos que tenemos.

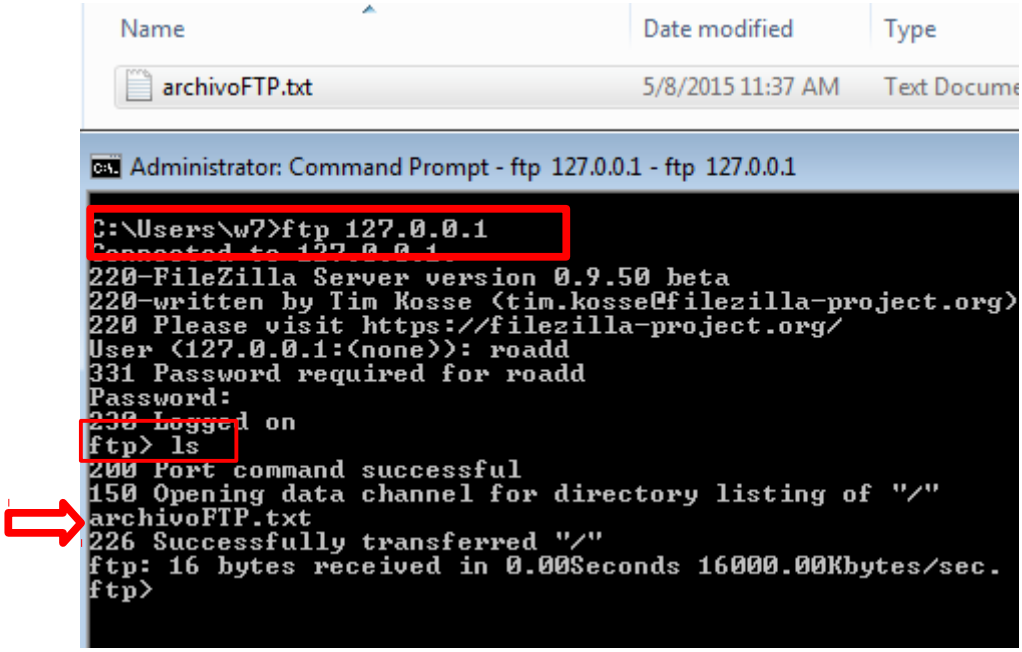
```
ftp> help
Commands may be abbreviated.  Commands are:

!               delete          literal         prompt          send
?               debug           ls              put             status
append          dir             mdelete         pwd             trace
ascii           disconnect      mdir            quit            type
bell            get             mget            quote           user
binary          glob            mkdir           recv            verbose
bye             hash            mls             remotehelp
cd              help            mput            rename
close           lcd             open            rmdir
ftp> _
```

Casi 45 comandos. No voy a enseñarles todos, pero sí algunos. Veamos una abreviación de lo que podemos entender rápidamente:

1. **cd:** Para ir navegando por los directorios. Tal cual el comando de Windows.
2. **Close o disconnect:** Desconectarse del server al cual está conectado sin salir del servicio.
3. **Bye o quit:** Cierra la conexión y el servicio.

4. **Rename <archivo> <nombre nuevo>:** renombra un archivo. Cuando se renombra es importante no olvidar la extensión del archivo.
5. **ls:** lista los archivos y directorios que hay en el directorio.
6. **Dir:** lista los archivos y directorios con sus atributos.
7. **Open <IP/DNS>:** Intenta abrir una conexión con un servidor.
8. **Pwd:** Pedirle al servidor que nos diga en qué directorio estamos. Cuando el directorio actual es "/" quiere decir que nos encontramos en la raíz, el primer directorio madre de todos los directorios (perdón la expresión, pero así se entiende xD).
9. **Status:** El estado de la conexión y sus parámetros.
10. **Verbose:** Cambia a On o a Off. La idea del verbose es que nos diga si queremos tener más información del server (On) o que las cosas pasen por segundo plano y no nos interese(Off).
11. **Delete <archivo>:** Borra un archivo.
12. **Rmdir <directorio>:** Borra un directorio.
13. **Get <archivo> [nombre nuevo]:** Transfiere un archivo desde el server ftp hacia donde estamos conectados nosotros. El segundo parámetro es opcional y es para ponerle un nuevo nombre cuando se termine de transferir.
14. **Mget <archivo> [archivo] … [archivo]:** Permite la transferencia de varios archivos simultáneamente
15. **Put <archivo>, Mput <archivo> [archivo] … [archivo]:** Igual que los anteriores pero desde la máquina host hacia el servidor FTP.
16. **!<comando>:** El signo de admiración más el comando es para ejecutar ese mismo comando en la máquina local.
17. **Binary y ascii:** Son modos de transferencia. El primero sirve para ejecutables, imágenes, mp3 -en fin, archivos binarios-. El segundo es para archivos de texto, html, y ese tipo de cosas.
18. **Mkdir <directorio>:** Crea un directorio.
19. **User:** Cambia el usuario que se conecta con el FTP.

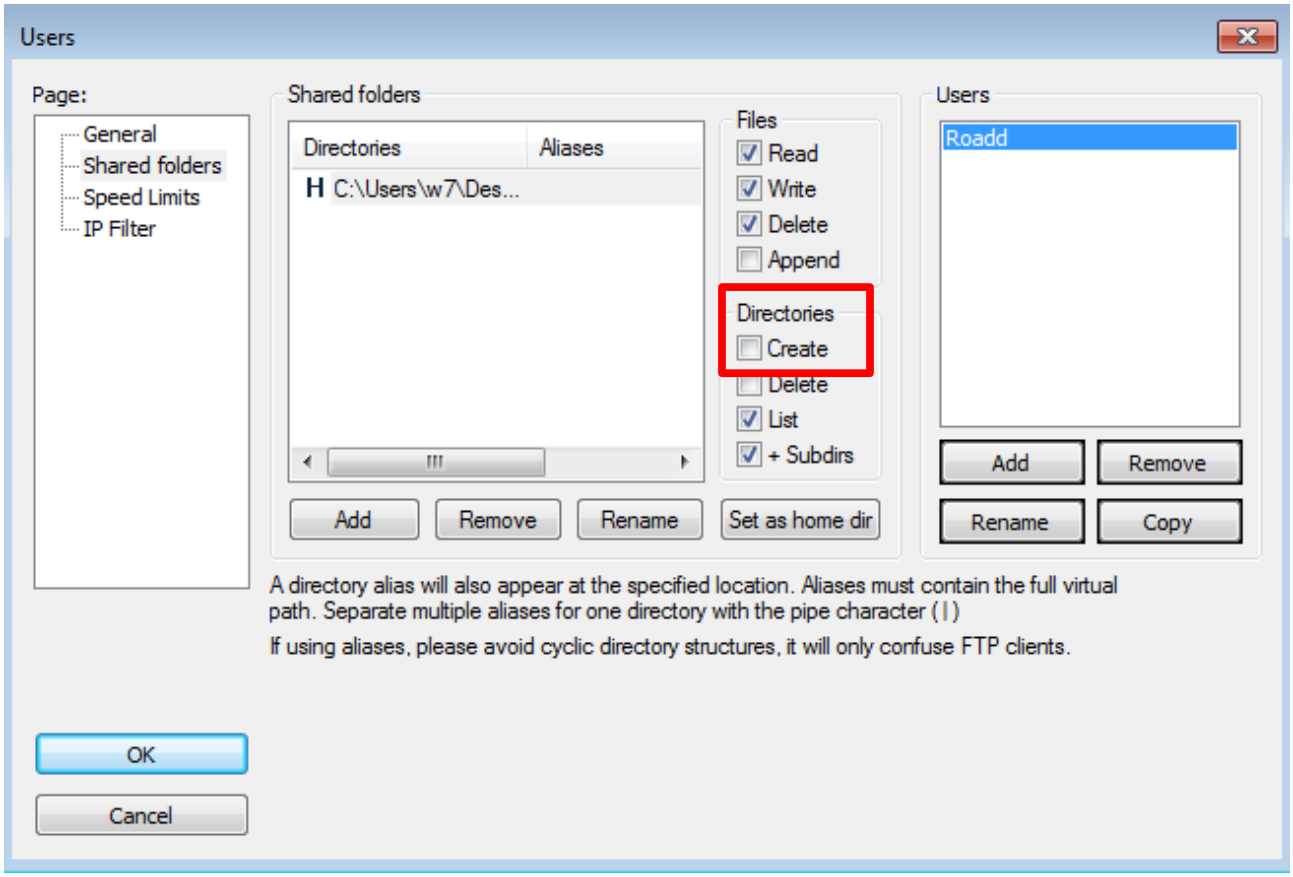
Sé que son comandos de oficina y administración, pasa que son necesarios. Vamos a ver la práctica:



Lo que aparece arriba es la carpeta "**madre**" -de ahora en más "**raíz**"-. Hicimos un ***ls*** que correspondía al **listado** de directorios y archivos. Nos dice también dónde listó ("/"). Voy a intentar crear un nuevo directorio.

```
ftp> mkdir nuevo
550 Can't create directory. Permission denied
ftp>
```

Pero **no tenemos permisos**. ¡Maldita sea la autoridad! ¡Anarquía! ¡Anarquía! Es broma, es broma. No se me duerman :P. Para que recuerden les vuelvo a mostrar la imagen donde no le pusimos **permisos de escritura**.



Voy a intentar **recuperar** el archivo de dentro del server.

```
ftp> get archivoFTP.txt
200 Port command successful
150 Opening data channel for file download from server of "/archivoFTP.txt"
226 Successfully transferred "/archivoFTP.txt"
ftp>
```

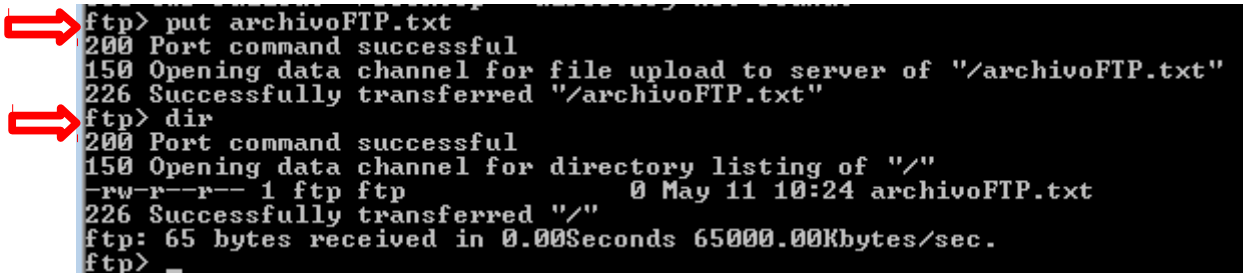
Y sí, todo salió bien :). Veamos si aparece en nuestras carpetas. Recuerden que era ***!*** para ejecutar un comando dentro de **nuestra máquina local**.

```
ftp> !dir
 Volume in drive C has no label.
 Volume Serial Number is 3403-DB3F

 Directory of C:\Users\w7

05/08/2015  12:57 PM    <DIR>
05/08/2015  12:57 PM    <DIR>
11/13/2014  09:52 AM
10/03/2014  02:33 PM    <DIR>
05/08/2015  12:57 PM                 0 archivoFTP.txt
01/28/2015  02:22 PM    <DIR>          Contacts
05/08/2015  11:37 AM    <DIR>          Desktop
08/20/2014  12:20 PM    <DIR>          Documents
```

Aparece perfectamente. Y ahora voy a **borrar** el archivo dentro del servidor.

```
ftp> delete archivoFTP.txt
250 File deleted successfully
ftp> ls
200 Port command successful
150 Opening data channel for directory listing of "/"
226 Successfully transferred "/"
ftp>
```

Perfecto. Todo bien. Veamos el tema de pasar un archivo hasta allí. Esto lo vimos en la segunda clase de malware pero nunca está de más volver a ver los temas. El comando ***put*** nos deja hacer la tarea.



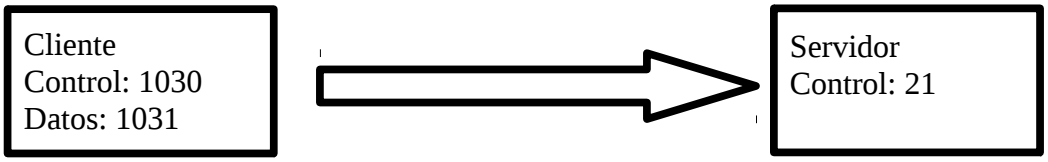
En rojo están marcados las líneas que corresponden a comandos.

Allí vemos los mensajes 226 correspondientes a las transferencias exitosas. Todo muy lindo.

"¿Y ésto es todo por el puerto 21? Es un super-puerto."

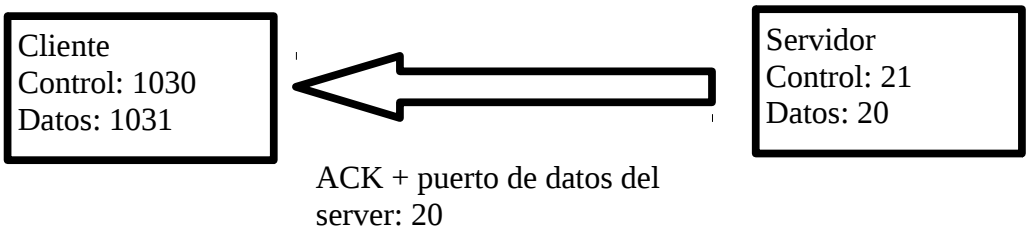
En realidad no todo pasa por el mismo puerto. **El puerto 21 se encarga del control** pero (siempre con algún pero, eh) la realidad es otra. Existen **dos formas de conexión: modo activo y modo pasivo**. Imagina que sino no podría haber más de un cliente conectado.

En el **modo activo** la historia es ésta: el puerto 21 del servidor está abierto esperando un mensaje. El cliente, con otro **puerto de control y otro de datos** de algún número mayor a 1024 -aleatorios- le dice al servidor FTP desde su puerto de control (digamos 1030) cuál es ese puerto de datos.

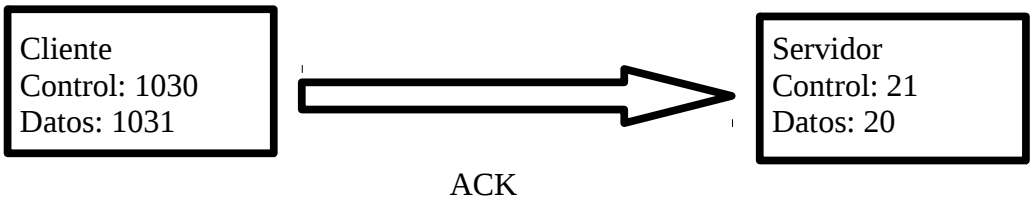


Servidor, quiero abrir una conexión de intercambio de datos. Mi puerto de datos es 1031.

La cursilería siempre hace todo más fácil de entender :P. Volvamos. Luego de ésto el server le responde con un ACK (recuerden el **three way handshake** de la conexión **TCP** que vimos casi al final de la **clase 17**) y envíando cuál es su puerto de datos.

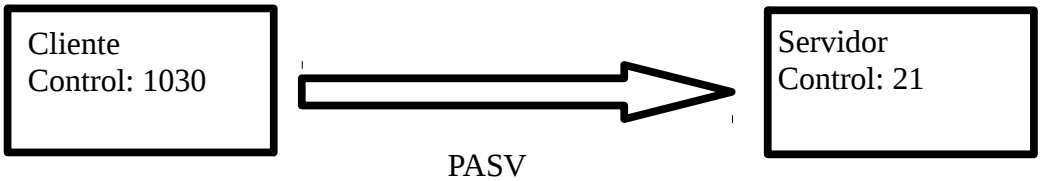


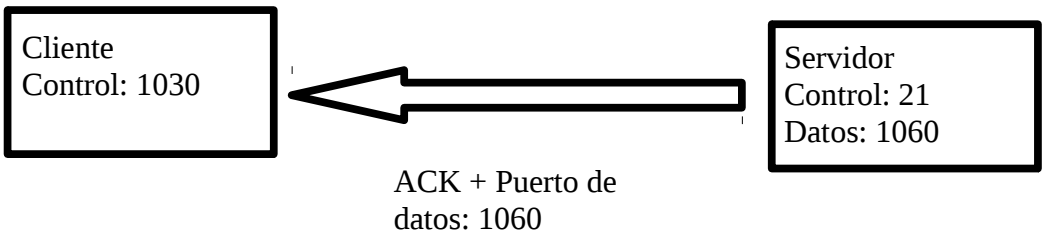
Si está todo bien, el cliente vuelve a responder al server con un **ACK**.

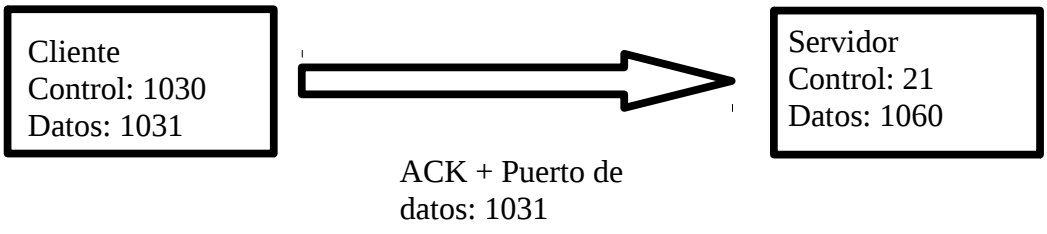


Aquí tenemos un grave **fallo de seguridad** porque el cliente le manda el puerto y el servidor se conecta a él. No vaya a ser cosa que alguien quiera anteponerse y hacerse pasar por el server. Por este problema -el **firewall** bloquea las conexiones de este estilo- se usa el **modo pasivo**.

El modo pasivo cambia porque primero le dice al servidor que quiere abrir una conexión mediante el comando **PASV** y el **servidor** le responde con el **canal de datos que usa**, que a diferencia del modo anterior debe ser mayor a 1024. Luego el cliente se conecta a él.



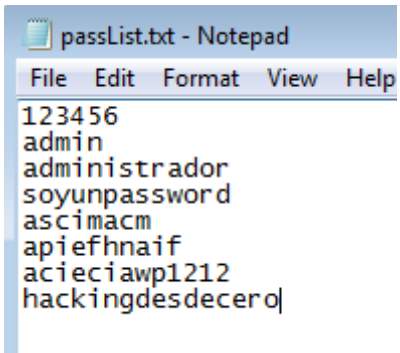




Aquí resolvemos este problema de seguridad, pero seguimos teniendo **otro**: todos los datos pasan en **texto plano** de un lugar a otro. **¿Qué pasa con el texto plano?** Que cualquiera que pueda interceptar los datos, al **no estar cifrados, se pueden leer sin inconvenientes**. Ya veremos como y de qué manera se puede arreglar. Por ahora tenemos que conformamos con hacerle **fuerza bruta**. Para ésto vamos a usar una herramienta que es muy buena en flexibilidad y en velocidad: **Hydra**. Les dejo el comprimido en los archivos complementarios, pero pueden googlearlo (ojo que también esta para Linux, así que los que usan Windows fíjense).

En fin, para usarlo tenemos que hacerlo desde la consola. Vayan hasta la carpeta donde está el **hydra.exe** y ejecútenlo.

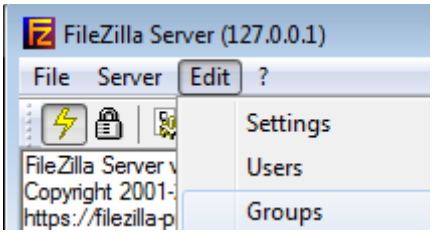
No sólo tenemos un buen modo de uso sino que el ejemplo es lo que queremos hacer. Voy a hacer un txt con contraseñas y **una será la correcta.** Ya se habrán dado cuenta que allí tenemos "**ftp://**" que es algo muy parecido a "**http://**" de cuando abrimos una página web. Bueno, ese antecesor a la dirección **especifica el protocolo**, que en este caso es ftp.

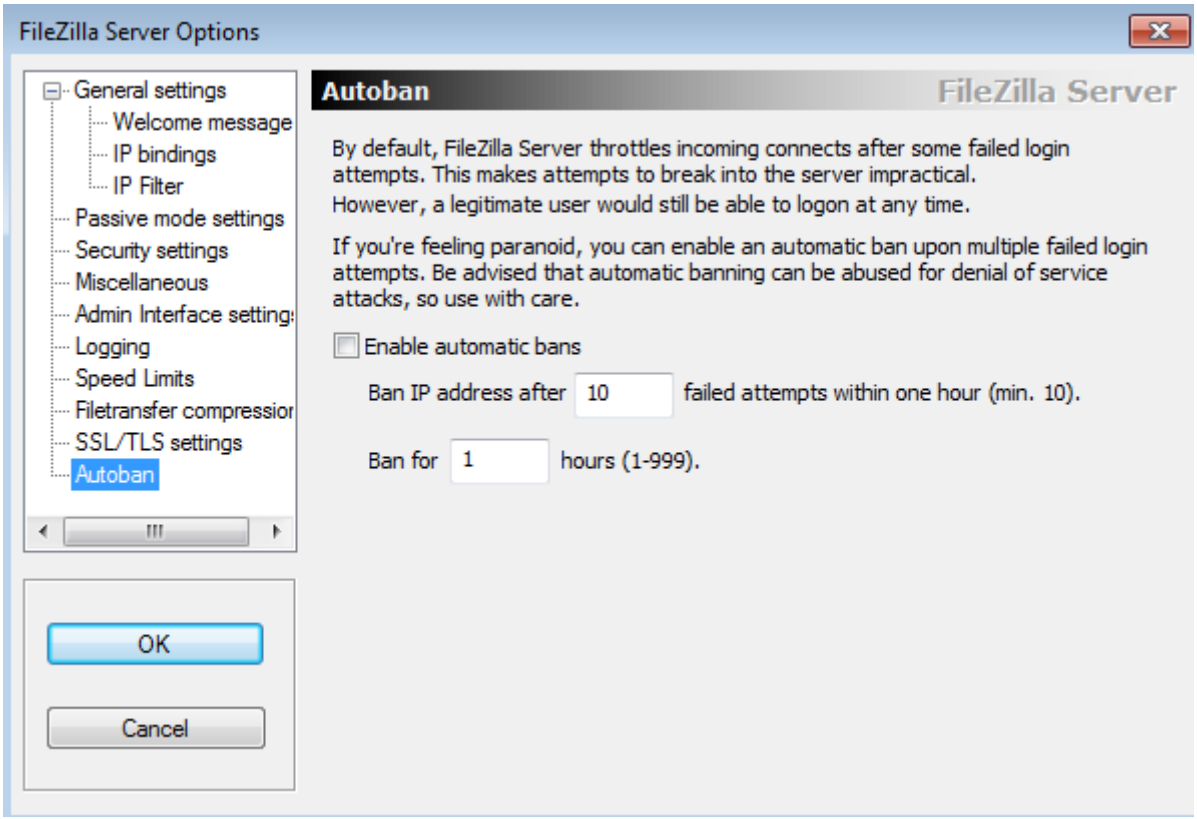


```
C:\Users\w7\Desktop\hydra-7.5\hydra>hydra -l roadd -P passList.txt ftp://127.0.0
.1
Hydra v7.5 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2015-05-11 13:15:49
[DATA] 8 tasks, 1 server, 8 login tries (l:1/p:8), ~1 try per task
[DATA] attacking service ftp on port 21
[21][ftp] host: 127.0.0.1   login: roadd   password: hackingdesdecero
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra)
```

Aquí vemos la información que nos tira y como encontró la **password correcta**. No hay mucho que decir aquí más que lo que ya sabemos :). **Investiguen la herramienta** que es excelente. **¿Y cómo hacemos para bloquear este tipo de ataques?**

Desde el Filezilla Server ***Edit*** → ***Settings*** y allí en la pestaña de ***Autoban***.





Si habilitamos la opción allí podemos configurar para que en el primer recuadro limitemos la cantidad de intentos **simultáneos**. **Si yerra 10 veces** (por como está configurado por default) tiene **1 hora de prohibición** para intentar poner 10 nuevas contraseñas. **Así nos ahorramos problemas -o nos lo generamos si nos olvidamos la clave xD-.**

Para la próxima clase veremos un servicio conocido como Telnet :). Espero les haya gustado.

Pueden seguirme en Twitter: @RoaddHDC

Cualquier cosa pueden mandarme mail a: r0add@hotmail.com

Para donaciones, pueden hacerlo en bitcoin en la dirección siguiente: 1HqpPJbbWJ9H2hAZTmpXnVuoLKkP7RFSvw

Roadd.
