Setting Up A Ftp:

//////////*1000+ HACKING TRICKS & TUTORIALS - ebook By Mukesh Bhardwaj Blogger - Paid Version - only @ TekGyd | itechhacks | Mukeshtricks4u*///////

Well, since many of us have always wondered this, here it is. Long and drawn out. Also, before attempting this, realize one thing; You will have to give up your time, effort, bandwidth, and security to have a quality ftp server.
That being said, here it goes. First of all, find out if your IP (Internet Protocol) is static (not changing) or dynamic (changes everytime you log on). To do this, first consider the fact if you have a dial up modem. If you do, chances are about 999 999 out of 1 000 000 that your IP is dynamic. To make it static, just go to a place like h*tp://www.myftp.org/ to register for a static ip address.

You'll then need to get your IP. This can be done by doing this:
Going to Start -> Run -> winipcfg or www.ask.com and asking 'What is my IP?'

After doing so, you'll need to download an FTP server client. Personally, I'd recommend G6 FTP Server, Serv-U FTPor Bullitproof v2.15 all three of which are extremely reliable, and the norm of the ftp world.
You can download them on this site: h*tp://www.liaokai.com/softw_en/d_index.htm

First, you'll have to set up your ftp. For this guide, I will use step-by-step instructions for G6. First, you'll have to go into 'Setup -> General'. From here, type in your port # (default is 21). I recommend something unique, or something a bit larger (ex: 3069). If you want to, check the number of max users (this sets the amount of simultaneous maximum users on your server at once performing actions - The more on at once, the slower the connection and vice versa).

The below options are then chooseable:
-Launch with windows
-Activate FTP Server on Start-up
-Put into tray on startup
-Allow multiple instances
-Show "Loading..." status at startup
-Scan drive(s) at startup
-Confirm exit

You can do what you want with these, as they are pretty self explanatory. The scan drive feature is nice, as is the 2nd and the last option. From here, click the 'options' text on the left column.

To protect your server, you should check 'login check' and 'password check', 'Show relative path (a must!)', and any other options you feel you'll need. After doing so, click the 'advanced' text in the left column. You should then leave the buffer size on the default (unless of course you know what you're doing), and then allow the type of ftp you want.

Uploading and downloading is usually good, but it's up to you if you want to allow uploads and/or downloads. For the server priority, that will determine how much conventional memory will be used and how much 'effort' will go into making your server run smoothly.

Anti-hammering is also good, as it prevents people from slowing down your speed. From here, click 'Log Options' from the left column. If you would like to see and record every single command and clutter up your screen, leave the defaults.

But, if you would like to see what is going on with the lowest possible space taken, click 'Screen' in the top column. You should then check off 'Log successful logins', and all of the options in the client directry, except 'Log directory changes'. After doing so, click 'Ok' in the bottom left corner.

You will then have to go into 'Setup -> User Accounts' (or ctrl & u). From here, you should click on the right most column, and right click. Choose 'Add', and choose the username(s) you would like people to have access to.

After giving a name (ex: themoonlanding), you will have to give them a set password in the bottom column (ex: wasfaked). For the 'Home IP' directory, (if you registered with a static server, check 'All IP Homes'. If your IP is static by default, choose your IP from the list. You will then have to right click in the very center column, and choose 'Add'.

From here, you will have to set the directory you want the people to have access to. After choosing the directory, I suggest you choose the options 'Read', 'List', and 'Subdirs', unless of course you know what you're doing . After doing so, make an 'upload' folder in the directory, and choose to 'add' this folder seperately to the center column. Choose 'write', 'append', 'make', 'list', and 'subdirs'. This will allow them to upload only to specific folders (your upload folder).

Now click on 'Miscellaneous' from the left column. Choose 'enable account', your time-out (how long it takes for people to remain idle before you automatically kick them off), the maximum number of users for this name, the maximum number of connections allowed simultaneously for one ip address, show relative path (a must!), and any other things at the bottom you'd like to have. Now click 'Ok'.
Requested

From this main menu, click the little boxing glove icon in the top corner, and right click and unchoose the hit-o-meter for both uploads and downloads (with this you can monitor IP activity). Now click the lightning bolt, and your server is now up and running.

Post your ftp info, like this:

213.10.93.141 (or something else, such as: 'f*p://example.getmyip.com')

User: *** (The username of the client)

Pass: *** (The password)

Port: *** (The port number you chose)

So make a FTP and join the FTP section

Listing The Contents Of A Ftp:

Listing the content of a FTP is very simple.
You will need FTP Content Maker, which can be downloaded from here:
ht*p://www.etplanet.com/download/application/FTP%20Content%20Maker%201.02.zip

1. Put in the IP of the server. Do not put "ftp://" or a "/" because it will not work if you do so.
2. Put in the port. If the port is the default number, 21, you do not have to enter it.
3. Put in the username and password in the appropriate fields. If the login is anonymous, you do not have to enter it.
4. If you want to list a specific directory of the FTP, place it in the directory field. Otherwise, do not

enter anything in the directory field.
5. Click "Take the List!"
6. After the list has been taken, click the UBB output tab, and copy and paste to wherever you want it.

If FTP Content Maker is not working, it is probably because the server does not utilize Serv-U Software.

If you get this error message:
StatusCode = 550
LastResponse was : 'Unable to open local file test-ftp'
Error = 550 (Unable to open local file test-ftp)
Error = Unable to open local file test-ftp = 550
Close and restart FTP Content Maker, then try again.

error messages:

110 Restart marker reply. In this case, the text is exact and not left to the particular implementation; it must read: MARK yyyy = mmmm Where yyyy is User-process data stream marker, and mmmm server's equivalent marker (note the spaces between markers and "=").
120 Service ready in nnn minutes.
125 Data connection already open; transfer starting.
150 File status okay; about to open data connection.
200 Command okay.
202 Command not implemented, superfluous at this site.
211 System status, or system help reply.
212 Directory status.
213 File status.
214 Help message. On how to use the server or the meaning of a particular non-standard command. This reply is useful only to the human user.
215 NAME system type. Where NAME is an official system name from the list in the Assigned Numbers document.
220 Service ready for new user.
221 Service closing control connection. Logged out if appropriate.
225 Data connection open; no transfer in progress.
226 Closing data connection. Requested file action successful (for example, file transfer or file abort).
227 Entering Passive Mode (h1,h2,h3,h4,p1,p2).
230 User logged in, proceed.
250 Requested file action okay, completed.
257 "PATHNAME" created.
331 User name okay, need password.
332 Need account for login.
350 Requested file action pending further information.
421 Too many users logged to the same account
425 Can't open data connection.
426 Connection closed; transfer aborted.
450 Requested file action not taken. File unavailable (e.g., file busy).
451 Requested action aborted: local error in processing.
452 Requested action not taken. Insufficient storage space in system.
500 Syntax error, command unrecognized. This may include errors such as command line too long.
501 Syntax error in parameters or arguments.
502 Command not implemented.
503 Bad sequence of commands.

504 Command not implemented for that parameter.
530 Not logged in.
532 Need account for storing files.
550 Requested action not taken. File unavailable (e.g., file not found, no access).
551 Requested action aborted: page type unknown.
552 Requested file action aborted. Exceeded storage allocation (for current directory or dataset).
553 Requested action not taken. File name not allowed.

Active FTP vs. Passive FTP, a Definitive Explanation

Introduction
One of the most commonly seen questions when dealing with firewalls and other Internet connectivity issues is the difference between active and passive FTP and how best to support either or both of them. Hopefully the following text will help to clear up some of the confusion over how to support FTP in a firewalled environment.

This may not be the definitive explanation, as the title claims, however, I've heard enough good feedback and seen this document linked in enough places to know that quite a few people have found it to be useful. I am always looking for ways to improve things though, and if you find something that is not quite clear or needs more explanation, please let me know! Recent additions to this document include the examples of both active and passive command line FTP sessions. These session examples should help make things a bit clearer. They also provide a nice picture into what goes on behind the scenes during an FTP session. Now, on to the information...

The Basics
FTP is a TCP based service exclusively. There is no UDP component to FTP. FTP is an unusual service in that it utilizes two ports, a 'data' port and a 'command' port (also known as the control port). Traditionally these are port 21 for the command port and port 20 for the data port. The confusion begins however, when we find that depending on the mode, the data port is not always on port 20.

Active FTP
In active mode FTP the client connects from a random unprivileged port (N > 1024) to the FTP server's command port, port 21. Then, the client starts listening to port N+1 and sends the FTP command PORT N+1 to the FTP server. The server will then connect back to the client's specified data port from its local data port, which is port 20.

From the server-side firewall's standpoint, to support active mode FTP the following communication channels need to be opened:

FTP server's port 21 from anywhere (Client initiates connection)
FTP server's port 21 to ports > 1024 (Server responds to client's control port)
FTP server's port 20 to ports > 1024 (Server initiates data connection to client's data port)
FTP server's port 20 from ports > 1024 (Client sends ACKs to server's data port)

In step 1, the client's command port contacts the server's command port and sends the command PORT 1027. The server then sends an ACK back to the client's command port in step 2. In step 3 the server initiates a connection on its local data port to the data port the client specified earlier. Finally, the client sends an ACK back as shown in step 4.

The main problem with active mode FTP actually falls on the client side. The FTP client doesn't make the actual connection to the data port of the server--it simply tells the server what port it is listening on and the server connects back to the specified port on the client. From the client side firewall this appears to be an outside system initiating a connection to an internal client--something that is usually blocked.

Active FTP Example

Below is an actual example of an active FTP session. The only things that have been changed are the server names, IP addresses, and user names. In this example an FTP session is initiated from testbox1.slacksite.com (192.168.150.80), a linux box running the standard FTP command line client, to testbox2.slacksite.com (192.168.150.90), a linux box running ProFTPd 1.2.2RC2. The debugging (-d) flag is used with the FTP client to show what is going on behind the scenes. Everything in red is the debugging output which shows the actual FTP commands being sent to the server and the responses generated from those commands. Normal server output is shown in black, and user input is in bold.

There are a few interesting things to consider about this dialog. Notice that when the PORT command is issued, it specifies a port on the client (192.168.150.80) system, rather than the server. We will see the opposite behavior when we use passive FTP. While we are on the subject, a quick note about the format of the PORT command. As you can see in the example below it is formatted as a series of six numbers separated by commas. The first four octets are the IP address while the second two octets comprise the port that will be used for the data connection. To find the actual port multiply the fifth octet by 256 and then add the sixth octet to the total. Thus in the example below the port number is ((14*256) + 178), or 3762. A quick check with netstat should confirm this information.

```
testbox1: {/home/p-t/slacker/public_html} % ftp -d testbox2
Connected to testbox2.slacksite.com.
220 testbox2.slacksite.com FTP server ready.
Name (testbox2:slacker): slacker
---> USER slacker
331 Password required for slacker.
Password: TmpPass
---> PASS XXXX
230 User slacker logged in.
---> SYST
215 UNIX Type: L8
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
ftp: setsockopt (ignored): Permission denied
---> PORT 192,168,150,80,14,178
200 PORT command successful.
---> LIST
150 Opening ASCII mode data connection for file list.
drwx------ 3 slacker users 104 Jul 27 01:45 public_html
226 Transfer complete.
ftp> quit
---> QUIT
221 Goodbye.
```

Passive FTP

In order to resolve the issue of the server initiating the connection to the client a different method for FTP

connections was developed. This was known as passive mode, or PASV, after the command used by the client to tell the server it is in passive mode.

In passive mode FTP the client initiates both connections to the server, solving the problem of firewalls filtering the incoming data port connection to the client from the server. When opening an FTP connection, the client opens two random unprivileged ports locally (N > 1024 and N+1). The first port contacts the server on port 21, but instead of then issuing a PORT command and allowing the server to connect back to its data port, the client will issue the PASV command. The result of this is that the server then opens a random unprivileged port (P > 1024) and sends the PORT P command back to the client. The client then initiates the connection from port N+1 to port P on the server to transfer data.

From the server-side firewall's standpoint, to support passive mode FTP the following communication channels need to be opened:

FTP server's port 21 from anywhere (Client initiates connection)
FTP server's port 21 to ports > 1024 (Server responds to client's control port)
FTP server's ports > 1024 from anywhere (Client initiates data connection to random port specified by server)
FTP server's ports > 1024 to remote ports > 1024 (Server sends ACKs (and data) to client's data port)

In step 1, the client contacts the server on the command port and issues the PASV command. The server then replies in step 2 with PORT 2024, telling the client which port it is listening to for the data connection. In step 3 the client then initiates the data connection from its data port to the specified server data port. Finally, the server sends back an ACK in step 4 to the client's data port.

While passive mode FTP solves many of the problems from the client side, it opens up a whole range of problems on the server side. The biggest issue is the need to allow any remote connection to high numbered ports on the server. Fortunately, many FTP daemons, including the popular WU-FTPD allow the administrator to specify a range of ports which the FTP server will use. See Appendix 1 for more information.

The second issue involves supporting and troubleshooting clients which do (or do not) support passive mode. As an example, the command line FTP utility provided with Solaris does not support passive mode, necessitating a third-party FTP client, such as ncftp.

With the massive popularity of the World Wide Web, many people prefer to use their web browser as an FTP client. Most browsers only support passive mode when accessing ftp:// URLs. This can either be good or bad depending on what the servers and firewalls are configured to support.

Passive FTP Example
Below is an actual example of a passive FTP session. The only things that have been changed are the server names, IP addresses, and user names. In this example an FTP session is initiated from testbox1.slacksite.com (192.168.150.80), a linux box running the standard FTP command line client, to testbox2.slacksite.com (192.168.150.90), a linux box running ProFTPd 1.2.2RC2. The debugging (-d) flag is used with the FTP client to show what is going on behind the scenes. Everything in red is the debugging output which shows the actual FTP commands being sent to the server and the responses generated from those commands. Normal server output is shown in black, and user input is in bold.

Notice the difference in the PORT command in this example as opposed to the active FTP example. Here, we see a port being opened on the server (192.168.150.90) system, rather than the client. See the discussion about the format of the PORT command above, in the Active FTP Example section.

```
testbox1: {/home/p-t/slacker/public_html} % ftp -d testbox2
Connected to testbox2.slacksite.com.
220 testbox2.slacksite.com FTP server ready.
Name (testbox2:slacker): slacker
---> USER slacker
331 Password required for slacker.
Password: TmpPass
---> PASS XXXX
230 User slacker logged in.
---> SYST
215 UNIX Type: L8
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> passive
Passive mode on.
ftp> ls
ftp: setsockopt (ignored): Permission denied
---> PASV
227 Entering Passive Mode (192,168,150,90,195,149).
---> LIST
150 Opening ASCII mode data connection for file list
drwx------ 3 slacker users 104 Jul 27 01:45 public_html
226 Transfer complete.
ftp> quit
---> QUIT
221 Goodbye.
```

Summary
The following chart should help admins remember how each FTP mode works:

Active FTP :
command : client >1024 -> server 21
data : client >1024 <- server 20

Passive FTP :
command : client >1024 -> server 21
data : client >1024 -> server >1024

A quick summary of the pros and cons of active vs. passive FTP is also in order:

Active FTP is beneficial to the FTP server admin, but detrimental to the client side admin. The FTP server attempts to make connections to random high ports on the client, which would almost certainly be blocked by a firewall on the client side. Passive FTP is beneficial to the client, but detrimental to the FTP server admin. The client will make both connections to the server, but one of them will be to a random high port, which would almost certainly be blocked by a firewall on the server side.

Luckily, there is somewhat of a compromise. Since admins running FTP servers will need to make their servers accessible to the greatest number of clients, they will almost certainly need to support passive FTP. The exposure of high level ports on the server can be minimized by specifying a limited port range for the FTP server to use. Thus, everything except for this range of ports can be firewalled on the server side. While this doesn't eliminate all risk to the server, it decreases it tremendously.