Ex030 Report

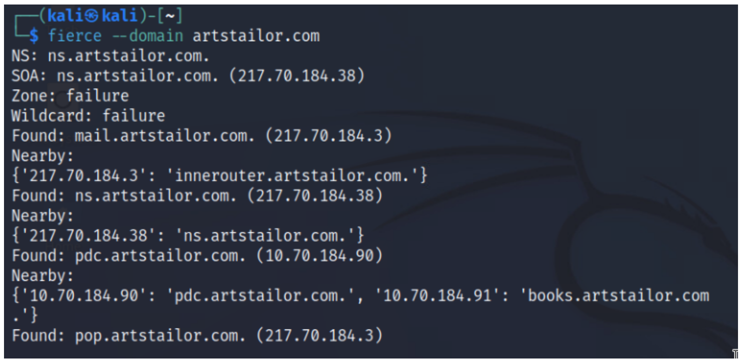
Benjamin Ruddy

2022-09-17

Contents

Attack Narrative

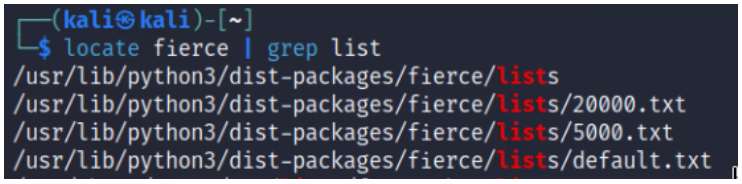
This exercise started with a simple DNS scan of `artstailor.com` with the `fierce` program. The discovered domains as well as their corresponding IPs can be seen in the following screenshot:



```
┌──(kali㉿kali)-[~]
└─$ fierce --domain artstailor.com
NS: ns.artstailor.com.
SOA: ns.artstailor.com. (217.70.184.38)
Zone: failure
Wildcard: failure
Found: mail.artstailor.com. (217.70.184.3)
Nearby:
{'217.70.184.3': 'innerouter.artstailor.com.'}
Found: ns.artstailor.com. (217.70.184.38)
Nearby:
{'217.70.184.38': 'ns.artstailor.com.'}
Found: pdc.artstailor.com. (10.70.184.90)
Nearby:
{'10.70.184.90': 'pdc.artstailor.com.', '10.70.184.91': 'books.artstailor.com
.'}
Found: pop.artstailor.com. (217.70.184.3)
```

Upon the output of the command, the source code of `fierce` was inspected to determine the location of the wordlist it uses. A brief scan of `/usr/bin/fierce` tells us that the file we are looking at is probably not the full code, and that an entry point is being loaded from a library that the program is using.

With the help of the `locate` command (after running `sudo updatedb`), we are able to list all files with the string `fierce` in them, and piping the output of that into `grep`, in which we filter for the term `list`, and ultimately find the wordlist that fierce uses by default:



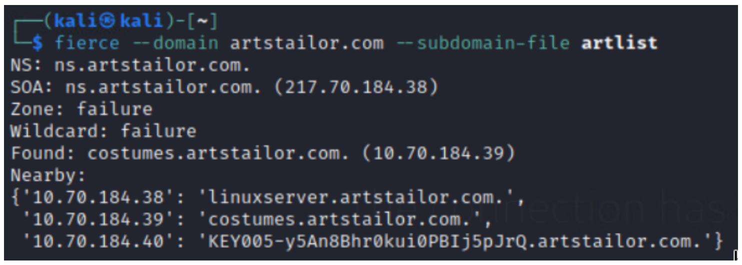
```
┌──(kali㉿kali)-[~]
└─$ locate fierce | grep list
/usr/lib/python3/dist-packages/fierce/lists
/usr/lib/python3/dist-packages/fierce/lists/20000.txt
/usr/lib/python3/dist-packages/fierce/lists/5000.txt
/usr/lib/python3/dist-packages/fierce/lists/default.txt
```

Our discovered domains that were found using this wordlist are: `ns`, `mail`, `pdc`, and `pop`.

Next, we attempt to expand upon our search by creating a custom wordlist with CeWL with the following command:

```
cewl http://www.artstailor.com -d 3 -o -w artlist
```

Having created our custom wordlist, we specify it with Fierce in order to find a total of three new hosts:



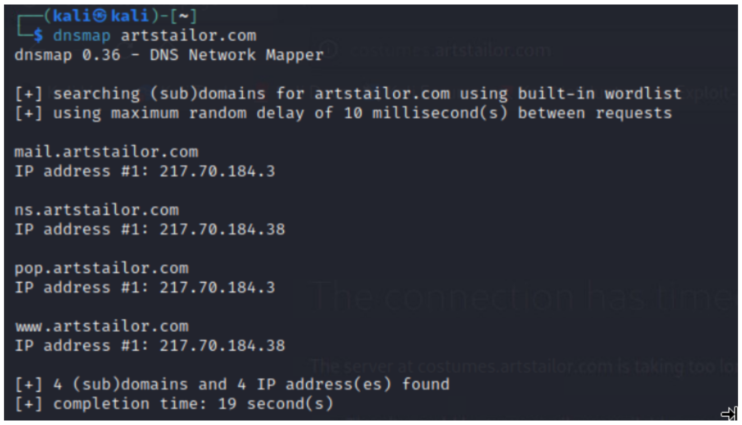
```
┌──(kali㉿kali)-[~]
└─$ fierce --domain artstailor.com --subdomain-file artlist
NS: ns.artstailor.com.
SOA: ns.artstailor.com. (217.70.184.38)
Zone: failure
Wildcard: failure
Found: costumes.artstailor.com. (10.70.184.39)
Nearby:
{'10.70.184.38': 'linuxserver.artstailor.com.',
 '10.70.184.39': 'costumes.artstailor.com.',
 '10.70.184.40': 'KEY005-y5An8Bhr0kui0PBIj5pJrQ.artstailor.com.'}
```

The following are the different methods that were used by `fierce` to identify our list of hosts:

- Usage of Fierce's default wordlist (default.txt)
  (the `subdomain_group.add_argument()` function is used here with the –subdomain-file option set to 'default.txt')
  - mail.artstailor.com
  - ns.artstailor.com
  - pdc.artstailor.com
  - pop.artstailor.com
- Using our custom wordlist
  - costumes.artstailor.com
- Using the nearby scan method (+-5 in the last octet for each host found; this can be adjusted with the –traverse option)
  - innerouter.artstailor.com (started from default wordlist)
  - books.artstailor.com (started from default wordlist
  - linuxserver.artstailor.com (started from our custom wordlist)
  - KEY005-y5An8Bhr0kui0PBIj5pJrQ.artstailor.com (started from our custom wordlist)

The above results could have indeed been obtained differently – By setting a higher traversal number, fierce would've checked a larger range of nearby IPs when it encountered the `pdc.artstailor.com` domain, revealing the three new hosts we got with CeWL if this number was set to 52 or higher.

Finally, below is the output of `dnsmap`, which found three hosts compared to `fierce`'s four:



```
┌──(kali㉿kali)-[~]
└─$ dnsmap artstailor.com
dnsmap 0.36 - DNS Network Mapper

[+] searching (sub)domains for artstailor.com using built-in wordlist
[+] using maximum random delay of 10 millisecond(s) between requests

mail.artstailor.com
IP address #1: 217.70.184.3

ns.artstailor.com
IP address #1: 217.70.184.38

pop.artstailor.com
IP address #1: 217.70.184.3

www.artstailor.com
IP address #1: 217.70.184.38

[+] 4 (sub)domains and 4 IP address(es) found
[+] completion time: 19 second(s)
```

MITRE ATT&CK Framework TTPs

There is one primary TTP that is applicable to our process in this exercise:

- **TA0043:** Reconnaissance
  - **T1590:** Gather Victim Networkk Information
    - **.002:** DNS