

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

The Assignment2 Summary:

*The write up contains the various techniques used corresponding to various lists (Yahoo, Linkedin, Formspring, XSplit), some Python code and output snippets
---- Techniques – Python Script, Hashcat – Dictionary Attack, Rule Attack, Hybrid Attack (Mask), Online Decryptor*

→ Comparison of Different Hashing and Storage:

- The difficulty in cracking are in the order of
 - **Yahoo (Plain Text) < Linkedin & XSplit (Raw Hash SHA1) < Formspring (Salted SHA256 Hash)**
- *The Raw Hash Password were relatively very easy to crack with many combinations of dictionaries. Online Decryptions were also possible to decrypt these passwords. Password cracking took very low time and high number of hits uncovered*
- *The Salted Hash was relatively a little difficult to crack to initially analyze pattern in which they were stored (Salt appended to them and the number of bits the salt were). Some weakly made password cracked from brute force attacks and from there on the cracking was relatively easy. Even then some strong passwords still require in depth use of some techniques to crack.*

→ YAHOO.txt – Plain text Password

- The Password file consisted of a lot of data and the password data was somewhere in the middle in the format
- **user_id : user_name : clear_passwd : passwd**

TECHNIQUE USED:

- **Python Script with Regex:**
 - Used Regular expression to extract the username and password in the format (Username <space> Password) as mentioned in the submission requirements
 - Python Script Used Below:

```
import re

def main():
    print "This Program Extracts the Plain Text Passwords from the Yahoo DB\n"
    print "Pattern : Yahoo: Plaintext , extract the plain text password from the file ! \n"
    print "Input Format --> user_id : user_name : clear_passwd : passwd\n\n"
    print "Output Format --> Username Password"

fd = open("/Users/darkknight/Desktop/Password Cracking/PasswordDumps/Password List/Yahoo.txt","r")
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
data = fd.readlines()
fd.close()

fd = open("/Users/darkknight/Desktop/Password Cracking/PasswordDumps/Yahoo_Extract.txt", "w")
for line in data:
    extract = re.search('^([0-9]+):(.*)$', line)
    if extract:
        if extract.group(1):
            if extract.group(2):
                fd.write("%s %s\n" % (extract.group(1), extract.group(2)))
            else:
                print ("\tFAIL: No Pattern Match for Username:Password")
    fd.close()

main()
```

- **Output:**

- This Program Extracts the Plain Text Passwords **from** the Yahoo DB

Pattern : Yahoo: Plaintext , extract the plain text password **from** the file !

user_id : user_name : clear_passwd : passwd

Output Format --> Username Password

Process finished **with** **exit** code 0

- **Output Snap: Attached file – Yahoo_Answers.txt**

ac1@associatedcontent.com @fl!pm0de@
john@associatedcontent.com pass
steveol@flash.net steveol
chotzi@aol.com chotzi
lb2512@yahoo.com lb2512
daveflomberg@yahoo.com scotch
jayschinderman@yahoo.com passwerd
leonardo.delarocha@gmail.com flipmode
miguel@associatedcontent.com flipmode
....

➔ **Linkedin.txt – SHA1 Hashed (no salt) Password**

- The file consisted of hashed password – hashed with sha1
- Identified the Hashes by the structure of the hash – length of the hash
- Brute forced with different hash types to confirm the identification of hash

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

- Read online of the hash type used by linkedin
“<https://arstechnica.com/security/2016/06/how-linkedins-password-sloppiness-hurts-us-all/>”

TECHNIQUE USED:

- **Python Script with Hashlib Module:**

The script basically uses hashlib module of python

- Reads the Hashes
- Reads the Dictionary for wordlist – (Used rockyou)
- Performs SHA1 Digest to the wordlist one by one
- Compare the hashed(wordlist) with the LinkedIn Hash
- Hit up if there is a match else continue – Write Matches to a file
- Python script used is found below

```
import hashlib,sys
def main():
    print " SHA1 Password Hashing Crack"
    print " LinkedIn: SHA1. Password = \"ISP\" then the hashed password is SHA1[\"ISP\"]"

    fd = open("/home/srinivas/Desktop/PasswordCrack/Password List/Linkedin.txt", "r")
    hashes = fd.readlines()
    fd.close()

    fd = open("/home/srinivas/Desktop/PasswordCrack/dict/dict1.txt", "r")
    data = fd.readlines()
    fd.close()
    count = 0

    ans = open("/home/srinivas/Desktop/PasswordCrack/answers.txt", "a")

    try:
        for hash in hashes:
            for dic in data:
                h = hashlib.sha1(dic.strip("\r\n")).hexdigest()
                if h == hash.strip("\r\n"):
                    print "Password Hit for %s\n" % (dic)
                    ans.write("%s %s\n" % (hash.strip(),dic.strip()))
                    count = count + 1

    except:
        print "Test Done !!! Count = %d" % (count)
    print "Test Done !!! Count = %d" % (count)
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
main()
```

-Output Sample

```
• /usr/bin/python2.7 /home/srinivas/PycharmProjects/PasswordCrack/linkedin.py
  SHA1 Password Hashing Crack
  Linkedin: SHA1. Password = "ISP" then the hashed password is SHA1["ISP"]
  Password Hit for ds323bcA

  Password Hit for 1KTMrallye

  Password Hit for Sub11endo!

  Password Hit for koelCopier

  Password Hit for Luddmilla99

  Password Hit for R00dpaard

  Password Hit for .linkedincvs.

  Password Hit for Myyefim01

  Password Hit for tech04chet

  Password Hit for cny4un123

  Password Hit for WINNIELINKEDIN

  Password Hit for tBiytc2009

  Password Hit for 6Xm3h2lq

  Password Hit for C0coA!16

  Password Hit for dory25143452

  Password Hit for woaizhangyi24
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
....  
Stopped the run after 3 hours
```

```
Test Done !!! Count = 667  
Test Done !!! Count = 667
```

-Output Snap – Password hit output stored in the separate file in format

<hash> <Password>

```
deb46f052152cfed79e3b96f51e52b82c3d2ee8e ds323bcA  
b3344eaec4585720ca23b338e58449e4c3d2f628 1KTMrallye  
674db9e37ace89b77401fa2bfe456144c3d2f708 Subllendo!  
37b5b1edf4f84a85d79d04d75fd8f8a1c3d2fbde koelCopier  
4f05e273b52ee943ab763d2bb3d83f5dc3d30904 Luddmilla99  
e417aded63377c45bbb7405edaa53d3cc3d30ba6 R00dpaard  
69a6dbf1bf05b16195eaf24f1fa43efdc3d317dd .linkedincvs.  
8145abd8e29dfe738096b117c771c538c3d319bb Myyefim01  
0861f15bc423e6690c3e7f456159fc07c3d31f03 tech04chet  
e69177b3636633b524162be07573abeec3d31fc0 cny4un123  
a4a48c1841161c01c44a9e68e8f11c5bc3d32078 WINNIELINKEDIN  
c6e173c0f381158c32f787e1d5c67530c3d32339 tBiytc2009  
4a9dcdb712976e2b43f34f72ed816503c3d33238 6Xm3h2lq  
3d1ddc5976823d85ae9a1e50d88ce161c3d33404 C0coA!16  
ba72a1f522016f4fd660fd19aa415ac5c3d33568 dory25143452  
2825562779ee1e9ab79beaaa6d64f356c3d33bf0 woaizhangyi24  
f2cc6e382acf33466e8160efc8c75d21c3d34520 aerdnaaerdnajd  
1d01d852fa4c1e5e82273e9c56afa612c3d35a98 hrcash7004  
8c064ffa2d6b7c205ce5010b4d2dbf25c3d36caf GRG$$ae0  
4b64869ec36d7a0d670dfac171dba42bc3d3722f Vmohanty85@  
bdb9a9e8a5ab18ec1ef3a7ca23276e9ec3d3871a abfbrkok43  
ebb49b3ab56f8fd49b472a40b69344eac3d3872b bpORG08*  
200dbd0023ef50c916a0cbd7a17e7504c3d39a53 PattyT723  
5edced403a4427956922e12d3d6e010bc3d39b45 OYEoye69  
d03f09fb5e7bdf5ab3f1c3df1ca436e8c3d3ab55 JustL00k1ng  
d10288e6acdcae6ae63d99543e858a8dc3d3b8b5 bns1tls2  
... More Included in the file attached
```

- **Online Decryptor – HashKiller (hashkiller.co.uk) –**
Used an online hash decrypter like the one mentioned hashkiller

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
We found 35 hashes! [Timer: 4028 m/s] Please find them below...

1f4e598963bef0e0ce48fb0b9c1faf265c3d73d4b SHA1 : CZ25666169
e65fa3077adb5b1ce070189a5d2f12dfc3d742e5 SHA1 : CZ25666169
000003077adb5b1ce070189a5d2f12dfc3d742e5 SHA1 : lemzhell1
0000068ce341cb0027941c52fa42f045c3d7430a SHA1 : lemzhell1
5b26a68ce341cb0027941c52fa42f045c3d7430a SHA1 : Jarv4040
314ebb31af8dd849ee61c2f6d2682b62c3d74439 SHA1 : winky1234
f9fd2b98b63268116a1fe2d04c052c5ac3d745ee [Not found]
00000b98b63268116a1fe2d04c052c5ac3d745ee SHA1 : Barco143
00000374c12c9484a904951cf43a888fc3d74ab9 SHA1 : linkedsw33ts
0000013a84f1c8766a1843d65bd6a6ec3d74b46 SHA1 : D&Kroatoan3001
48830e3bea1684e8bb92ee9183929fb0c3d74b92 SHA1 : D&Kroatoan3001
00000b8744cd66fa2677d43c73457e2c3d74c5a SHA1 : spid3rcr4b
712b0dbb23b015b5db2c84b04928bbac3d757c4 SHA1 : spid3rcr4b
00000fbbac8cd727b574820201d53e48c3d75ed2 SHA1 : MoFarr01
000002cfbee49f2129ccca633580b0e5c3d76661 SHA1 : viola6libera
0000088d2efef11ce877d939978edf62c3d76874 SHA1 : viola6libera
000003bd87c00a55f66ac5dc40001dfc3d76b5c SHA1 : 3495763
00000c4e1e8f5d1b883957f637a5f153c3d76dde SHA1 : mattiviskarl
34cb41f3a49b0ce527bd129eaf4111f3c3d76df7 SHA1 : _0m010k4j
0000059c0ff46296acf3141a8f0888e6c3d76dff SHA1 : 37123712
cc88d8ab3d82cda3c666ea8a165f06d0c3d77934 [Not found]
0fa15e948a39865a935df671928c2835c3d779bd SHA1 : ss063077
f060fa068a1ae7bd56a15bc71be7804ec3d77a25 SHA1 : trigger66
6926bfcddd388cc6e10e702575c6edc0c3d7846d SHA1 : powergame
00000603447e90aff44acd1132dd7d12c3d785b0 SHA1 : thetec7
0000046a038190184f8b098b02979fa8c3d78957 SHA1 : naddoune
000005822062cd044a39585079848a6c3d78a63 SHA1 : esofmcA
00000fba9f404daaf29d708ed3bfe57fc3d78b6a SHA1 : germany2006
4e7c6fba9f404daaf29d708ed3bfe57fc3d78b6a SHA1 : j, hp. 56
7e2318e2a8cb93488350387cb1546a62c3d79256 [Not found]
00000fa51761ae213aa2a7a85a43b1f5c3d7980c SHA1 : sArAng1729
1a6f7ee24c142c14c0561ae21c8b9c9ac3d79943 [Not found]
e82748f90bc81aa56911b8a6ce56d652c3d79aac SHA1 : 83669680
d2d326d22bcde27b75cd674ec8658308c3d79f54 SHA1 : Liquid76
000006d22bcde27b75cd674ec8658308c3d79f54 SHA1 : mile70
8009123a7d611d5c2704ab7167c91554c3d79f7 SHA1 : Hardyisland12
4e2318e2a8cb93488350387cb1546a62c3d79256 SHA1 : Hardyisland12
7e2318e2a8cb93488350387cb1546a62c3d79256 SHA1 : Matchbold09
00000fa51761ae213aa2a7a85a43b1f5c3d7980c SHA1 : stupid2me
1a6f7ee24c142c14c0561ae21c8b9c9ac3d79943 [Not found]
```

- **Hashcat – Used Hashcat Tool with Kali Linux**

- As we have prior knowledge of the hash used (SHA1), use Hashcat with the hash type set to SHA1 and made a **DICTIONARY Attack** with rockyou and other available wordlist at

http://hashcrack.blogspot.de/p/wordlist---ds_29.html and

<http://www.skullsecurity.org/wiki/index.php/Passwords>

- **Command:**

```
hashcat -m 100 --potfile-disable Linkedin.txt ../rockyou.txt
```

- **Output: Aborted in few seconds to capture the output**

hashcat (v3.10) starting...

Counting lines in Linkedin.txt

Parsed Hashes: 0/6143150 (0.00%)

Parsed Hashes: 131072/6143150 (2.13%)

Parsed Hashes: 262144/6143150 (4.27%)

Parsed Hashes: 393216/6143150 (6.40%)

Parsed Hashes: 524288/6143150 (8.53%)

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

...

Parsed Hashes: 6029312/6143150 (98.15%)

Parsed Hashes: 6143150/6143150 (100.00%)

Removing duplicate hashes...

Structuring salts for cracking task...

Generating bitmap tables with 16 bits...

Generating bitmap tables with 23 bits...

OpenCL Platform #1: Intel(R) Corporation

=====

- Device #1: Intel(R) Core(TM) i5-6360U CPU @ 2.00GHz, 501/2004 MB allocatable, 2MCU

OpenCL Platform #2: Mesa, skipped! No OpenCL compatible devices found

Hashes: 6143150 hashes; 5545381 unique digests, 1 unique salts

Bitmaps: 23 bits, 8388608 entries, 0x007fffff mask, 33554432 bytes, 5/13 rotates

Rules: 1

Applicable Optimizers:

** Zero-Byte*

** Precompute-Init*

** Precompute-Merkle-Demgard*

** Early-Skip*

** Not-Salted*

** Not-Iterated*

** Single-Salt*

** Raw-Hash*

Watchdog: Temperature abort trigger disabled

Watchdog: Temperature retain trigger disabled

Initializing device kernels and memory...

Checking for weak hashes...

Cache-hit dictionary stats ../dict1.txt: 193576069 bytes, 15844218 words, 15844218 key-space

[s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit =>

000008184569d68359358ff314765c82166f9dfd:!!!!!!

[s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit =>

0000010b57694951c5ca9405c741fcc7578af9b1:!!!!!!

[s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit =>

000008f0d07e0aa59dd56eb01766f434e74d02c0:!!QAZ22wsx

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
[s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit =>
```

```
00000bdc7f697e939c7d9fa0f14e0d1568e0c1ec:!!QAZxsw22
```

```
[s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit =>
```

```
00000a65ecc0df57676ecc826733c820555a1548:!!URdead
```

```
[s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit =>
```

```
00000334bad2aea5d60f6b251fe6ff678b59dcab:!!Will92
```

```
[s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit =>
```

...

INFO: approaching final keyspace, workload adjusted

```
[s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit =>
```

Session.Name...: hashcat

Status.....: Exhausted

Input.Mode.....: File (.././../dict1.txt)

Hash.Target.....: File (Linkedin.txt)

Hash.Type.....: SHA1

Time.Started...: Sat Feb 4 17:43:35 2017 (22 secs)

Speed.Dev.#1...: 2839.5 kH/s (0.47ms)

Recovered.....: 628673/5545381 (11.34%) Digests, 0/1 (0.00%) Salts

*Recovered/Time.: CUR:N/A,N/A,N/A AVG:1713159.00,102789544.00,2466949120.00
(Min,Hour,Day)*

Progress.....: 15844218/15844218 (100.00%)

Rejected.....: 222958/15844218 (1.41%)

Started: Sat Feb 4 17:43:35 2017

Stopped: Sat Feb 4 17:44:20 2017

→ Formspring.txt – SHA1 Hashed (with 2 Digit salt)

Password

- The file consisted of hashed password – hashed with SHA256
- Identified the Hashes by the structure of the hash – length must be > SHA1
- Read Online at http://www.theregister.co.uk/2012/07/11/formspring_security_breach/ to confirm analysis

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

- Made a Dictionary Attack with Hash cat – Brute Forced for the Hash Type
- Got Few Outputs which helped to analyze the pattern to be Hash(Password + 2Digit Salt)
- Matched the same structure with the Hashes (from hash file) too, which end in the almost similar fashion
- Used Hash cat to make a **RULE Attack** and a **HYBRID Attack**
- Python Script with SHA256 Salted Hash Matching with Dictionary – Uncovered few but took a long time (Iterated from 00 – 99 salts each time)
- Tried to use PyCuda to use the GPU but have setup issues with drivers in my machine.
- **HASH CAT – DICTIONARY ATTACK** – To get the speed results and crack few passwords to recognize the pattern
 - Executed Hashcat Dictionary attack brute forcing for the Hash value starting from SHA-1 to SHA256 --- Hit encountered at SHA256 for few entries
 - This helped in analyzing the pattern of the password stored in the file
 - The Format was – HASH (2 Digit Salt + Password)
 - Got “Line Length Exception” for SHA-1 and others

Output Below:

- **Output 1** – Dictionary Attack with SHA256

```
root@kali:~/Desktop/PasswordCrack/isp/Password List/Password List# hashcat -m 1400 --potfile-disable formspring.txt ../../dict1.txt
hashcat (v3.10) starting...
```

```
OpenCL Platform #1: Intel(R) Corporation
```

```
=====
```

```
- Device #1: Intel(R) Core(TM) i5-6360U CPU @ 2.00GHz, 501/2004 MB
allocatable, 2MCU
```

```
OpenCL Platform #2: Mesa, skipped! No OpenCL compatible devices found
```

```
Hashes: 419564 hashes; 419564 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1
Applicable Optimizers:
* Zero-Byte
* Precompute-Init
* Precompute-Merkle-Demgard
* Early-Skip
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

- * Not-Salted
- * Not-Iterated
- * Single-Salt
- * Raw-Hash

Watchdog: Temperature abort trigger disabled
Watchdog: Temperature retain trigger disabled

Cache-hit dictionary stats ../../../../dict1.txt: 193576069 bytes, 15844218 words, 15844218 keyspaces

f2f96a735b6255fb0d29950e92a47ba033171b36443aeeb28e8e2c90cd12f3dc:552blueeyes
a587a5f8d3ed8c98a5707e346103f84561d2b446c719f3cf30ffcf6b483b752:666naruto
ce2ac806570a69baf963553347385a74ee8266e79cde68d9dfbfef77e3a9bb80:83chs2010
INFO: approaching final keyspaces, workload adjusted

Session.Name....: hashcat
Status.....: Exhausted
Input.Mode.....: File (../../../../dict1.txt)
Hash.Target....: File (formspring.txt)
Hash.Type.....: SHA256
Time.Started...: Thu Feb 9 02:32:42 2017 (6 secs)
Speed.Dev.#1...: 2206.6 kH/s (0.68ms)
Recovered.....: 3/419564 (0.00%) Digests, 0/1 (0.00%) Salts
Recovered/Time.: CUR:N/A,N/A,N/A AVG:29.39,1763.35,42320.39 (Min,Hour,Day)
Progress.....: 15844218/15844218 (100.00%)
Rejected.....: 222958/15844218 (1.41%)

Started: Thu Feb 9 02:32:42 2017
Stopped: Thu Feb 9 02:32:51 2017

- Output2:

Dictionary attack with rockyou Dictionary

root@kali:~/Desktop/PasswordCrack/isp/Password List/Password List# hashcat -m
1400 --potfile-disable formspring.txt ../../rockyou.txt
hashcat (v3.10) starting...

OpenCL Platform #1: Intel(R) Corporation

=====

- Device #1: Intel(R) Core(TM) i5-6360U CPU @ 2.00GHz, 501/2004 MB
allocatable, 2MCU

OpenCL Platform #2: Mesa, skipped! No OpenCL compatible devices found

Hashes: 419564 hashes; 419564 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable Optimizers:

- * Zero-Byte
- * Precompute-Init
- * Precompute-Merkle-Demgard
- * Early-Skip
- * Not-Salted
- * Not-Iterated
- * Single-Salt

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

* Raw-Hash

Watchdog: Temperature abort trigger disabled

Watchdog: Temperature retain trigger disabled

Cache-hit dictionary stats ../../rockyou.txt: 139921497 bytes, 14343296 words, 14343296 keyspaces

0ec2cc1ad2208506118f565020743221bd7c1db2b7b1d69bea8fdcf6c8d2085d:9511coolie
a587a5f8d3ed8c98a5707e346103f84561d2b446c719f3cf30ffcf6b483b752:666naruto
485d27fe2bcfeb5ff1c16c1abfb01b8aad985d1c2e0f31ee518cfc00facf84fa:19brittany94
34ab2408e785e72e41b42b4d781c1950e6867d79dc9c4e43fd2fc89a50f1b46f:12tasty85
72e9474b0555e01c22c389315a760e383302eb6698152446d3685fd45c58df8b:0123nikki
INFO: approaching final keyspaces, workload adjusted

Session.Name....: hashcat
Status.....: Exhausted
Input.Mode.....: File (../../rockyou.txt)
Hash.Target....: File (formspring.txt)
Hash.Type.....: SHA256
Time.Started...: Thu Feb 9 02:34:05 2017 (6 secs)
Speed.Dev.#1...: 2111.6 kH/s (0.66ms)
Recovered.....: 5/419564 (0.00%) Digests, 0/1 (0.00%) Salts
Recovered/Time.: CUR:N/A,N/A,N/A AVG:49.98,2999.04,71976.99 (Min,Hour,Day)
Progress.....: 14343296/14343296 (100.00%)
Rejected.....: 1599/14343296 (0.01%)

Started: Thu Feb 9 02:34:05 2017

Stopped: Thu Feb 9 02:34:14 2017

- **Python Script – Salted SHA256 Hash ---- Create a file with 2 Digit Salt (00-99) and iterate each wordlist by concatenating the salts before hashing**

The Code flow is as follows,

- Read the Hashes and the Wordlist (rockyou) and store in respective variables
- Create a Salt file with the required 2 digit salts as analyzes (00-99)
- Iterate the hashes and for each hash, use a word from list and hash it with salts from 00 to 99 and match them
- If match occurs return the combination and write them into answer file

```
import hashlib,sys
def main():
    print " Formspring Password Hashing Crack"
    print "" Formspring Hashes - SHA256 with 2 Digit Salt - SHA256(salt+Password)""

    fd = open("/home/srinivas/Desktop/PasswordCrack/test.txt", "r")
    hashes = fd.readlines()
    fd.close()
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
fd = open("/home/srinivas/Desktop/PasswordCrack/test1.txt", "r")
data = fd.readlines()
fd.close()
count = 0

fd = open("/home/srinivas/Desktop/PasswordCrack/salt.txt", "a")
fd.write('00\n')
fd.write('01\n')
fd.write('02\n')
fd.write('03\n')
fd.write('04\n')
fd.write('05\n')
fd.write('06\n')
fd.write('07\n')
fd.write('08\n')
fd.write('09\n')
for i in range(10,100):
    i = str(i) + "\n"
    fd.write(i)
fd.close()

# sys.exit()
fd = open("/home/srinivas/Desktop/PasswordCrack/salt.txt", "r")
salt = fd.readlines()
fd.close()
count = 0

ans = open("/home/srinivas/Desktop/PasswordCrack/answers.txt", "a")

try:
    for hash in hashes:
        for dic in data:
            #SALT CODE
            for s in salt:
                #print s.strip('\r\n') + dic.strip('\r\n')
                sh = s.strip() + dic.strip()
                h = hashlib.sha256(sh).hexdigest()
                if h == hash.strip('\r\n'):
                    print "Password Hit for %s\n" % (sh)
                    ans.write(dic)
                    count = count + 1
except:
    print "Test Done !!! Count = %d" % (count)
    print "Test Done !!! Count = %d" % (count)

main()
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

Output is as follows – Stopped after 15+ encounters as the script run takes a long time

```
/usr/bin/python2.7 /home/srinivas/PycharmProjects/PasswordCrack/formspring.py
```

Formspring Password Hashing Crack

Formspring Hashes - SHA256 with 2 Digit Salt - SHA256(salt+Password)

Password Hit for 94zoomster

Password Hit for 6500cruiser

Password Hit for 8200lighting

Password Hit for 0600love00

Password Hit for 6400purple!

Password Hit for 26010203la

Password Hit for 73010886k

Password Hit for 66012wrestbg

Password Hit for 660147asdf

Password Hit for 130161563;

Password Hit for 2101dontno

Password Hit for 410202023a

Password Hit for 88021896h

Password Hit for 910230898b

Password Hit for 8602steve

Password Hit for 940305767k

Password Hit for 33030701park

Test Done !!! Count = 17

Process finished with exit code 0

- **HASH CAT – HYBRID ATTACK with mask ?d?d (2 Digit Numbers)**
 - Executed a Hybrid Mask + Dictionary attack
 - Applied the Mask before the password in the wordlist
 - Flags used : SHA256 (-m 1400) and Hybrid Mode (-a 7 ?d?D)
 - Tried for both Mask append to front and back of wordlist

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

- This technique cracked numerous passwords in the list

Output: Aborted in few seconds to record the output

```
root@kali:~/Desktop/PasswordCrack/isp/Password List/Password List#  
hashcat -m 1400 --potfile-disable formspring.txt -a 7 ?d?d  
../../../../dict1.txt  
- hashcat (v3.10) starting...  
-  
- OpenCL Platform #1: Intel(R) Corporation  
- =====  
- - Device #1: Intel(R) Core(TM) i5-6360U CPU @ 2.00GHz, 501/2004 MB  
  allocatable, 2MCU  
-  
- OpenCL Platform #2: Mesa, skipped! No OpenCL compatible devices found  
-  
- Hashes: 419564 hashes; 419564 unique digests, 1 unique salts  
- Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13  
  rotates  
- Applicable Optimizers:  
- * Zero-Byte  
- * Precompute-Init  
- * Precompute-Merkle-Demgard  
- * Early-Skip  
- * Not-Salted  
- * Not-Iterated  
- * Single-Salt  
- * Raw-Hash  
- Watchdog: Temperature abort trigger disabled  
- Watchdog: Temperature retain trigger disabled  
-  
- Cache-hit dictionary stats ../../../../dict1.txt: 193576069 bytes,  
  15844218 words, 1584421800 keyspaces  
-  
- 57f76df0c45a161bf9eb1e591518e3359a5d73425979fa293f2f29ef4c4c6821:98!!p  
  assword  
- 49096c2552120814aa09704baa97430ea7e616df18ab5f2ad91b0b61d875fc4e:62!1k  
  imchi  
- 60cae5cdde7cc36b6e95c835383db8d047b248355ed3004821e50d74e2205603:62!@#  
  QWEASD  
- 108f6ae5888c4cf1899b8559056ef98666606bb427745d97ed7fdd654224aa1e:08!ha  
  wthorne!1  
- ccea52791a7f5394ce193641785b67721008699e1d30a4396a66d7cb11550b15:96!il  
  oveyou!  
- 1134362d3c422d03b420d64db3124aeb52c6d89672e29604aefeffc5609ca1:46!re  
  tep  
- c6645085e4ce9bfd250388bb077b610ff98848ecb397255cea946fd2e2ffa960:79"fa  
  izan786"  
- d4b63e091f22d55b15cbf2915585589496cd7888d2eb872497dd696d70b8b6ca:98##e  
  ric$$  
- c48e94c672d0e767ac82a08ab66fc9cbfa825ac6b50a81b8d35c1bbad3b6e90a:76#1b  
  iker
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

- 8049cc1b77312888c27cdb2b43673e33bc53293ce27e55d2cb3af25cc7d49d22:73#1c
hick
- 3113ea8be7fd50fd9bc7688e92354682aebbd2a2bfb19e7b9e9b2cbdeaac8ed2:16#1c
hick
- 4e4a117d504c13678eadf3bca2934c65e18cc22c500ac5b718a16ad8efb4e22f:35#1p
reston
- 93bda05dca68a6f696e6b0a80bad92e77db28d7cb33a2d6944fb125f431f3945:86#1s
hopper
- 2308f38415e9bbea5f077c85bf2bc4147d0d6f732e8172365f14060f280850b3:01#1s
inger
- 0fd405f18e3123a99c8be6e85709c351441d9bead3706496a83b6ecbd9550c6d:83#1s
inger
- d51161fe7800e417dbc97e1628ea152f1de92f50a707bcd9d56a7fcba3ac57a5:48#2f
unny
- d170f27a3ed040137f5deedc1c68c413ae873da8062e61c066a796bb5d807951:96\$ \$m
oney
- 6220a316b7b5c49f22aac29e4514b98202e629c1cd2619707b62f63ae02007e7:71\$3c
uri73
- f62c824fbe0578ece5d38ae380a57466311696a336e099dcb846c2be51c2fcbc:88&he
arts;
- 45f6404f421675d7364a38381fe223f67643224cef79e8573c4497c1d19618e8:01&he
arts;
- e2d432c3e0da472f4f8147062f8d0f74c881cf95eaa6a60750ede1850b57a363:78&he
arts;
- 689b64362aed337dceff20e4f937ab4fa953f8c646bab2e79f9cee7c63a8594a:54&he
arts;
- 34475d6dc625cde4118f57c94912fa8b67b768cc14bc5aa36882ae1e47b63a72:08&he
arts;
- 8e2587293e2866dd8c2789df9697b22c8cf30209ebd6e75d95107e9309793a7f:35&he
arts;
- 38e743c809473719338ab95181320050ecce2928f9c890df1105a24d67b83c42:53&he
arts;
- d8656eaa255a13ad597fbd84bab7a5a6f11b9b30f584d903ea21c4d9554babf6:61&he
arts;
- ebd3a983f98eb36fecfcf00afac65ab5932ef71cd8e6c6e006c670e14fc92d26:79&he
arts;
- 462a8606439581b5068059dd97f79af6376794b3f366e110d040664ad1222fe4:18&he
arts;
- 13221a092cb3cfeba5475218c8c7aa2f2c783f7b64b9f334980b5e7008c95d62:81&he
arts;
- b1fd9a45b9ff7327f32a5bc954e5f7cd25e77b35679100b0c86e5d61b96c6df0:46 (se
renity)
- caabc7ca475df21b7f6adc4e72057737b50ea93f40afb0f1694b81a81d2d8e94:14*82
5361910
- f9c6f429575f99ae10744a3c3451df6a9bfb2a4640f3b59e03b0a5e6cdb210df:15*cu
pcake
- 28f4aef67d8eae43ba9b1830b68318a1135d5db95f38b7b1d433c9b78d19dcc1:27*b1
ackie
- 35ecab6d5f14ca0f77cfa367973a61cad487f9754c6382749396f867feb0cc8:75*ey
ahsed*
- 7eafdbadab8f967292e36e5bcd97ca317d3d45b90026b99e4461809647736d35:72-
angel66-

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
- 7d23c5cb801df546835c91c9ca4be04c336d1d0b09f577dcec5600fd7805634f:89.45
  slug
- 748a3de2b41a72d34bbac6ffd0c7ad7da5dfebd7648fced136fc108ca5949b07:56.ki
  tty.
- ec9f10079e9445e5a11b43441c16f368ea010d523e9c1a7d3426a3b814e06722:45.or
  ion.
- 809224bee73cd38234a41db656c3ec91a7e02f64e90adc59dd9592f924b5c241:42.st
  art.
- 6a3a58f989635bccda12e7945acbe46e4912c4c568252cd26a68ab1479857191:92000
  278865a
- 5a62f430c05576991fcb53a80878400cdd52076614e26b714e2da9d9c335724b:30000
  hlala
- 8ed5ee104e7dfaa5799d05e0664d61f7df54d9701aa65fe9ffe123ef13a6be3c:54000
  money
- 1ed341745fab8abab2e5394fd82acfca5046d98a0ba5006596fde3db202dee4e:03005
  231a
- e9f41c4a6607aca4566e818a9b03f1ee33fb729e4d5dde7dbdbec382de143df:40007
  ndyup
- a214beb612a41c947dd6a405d87acb786d9c2660c148f4c241914a7f994f8f0b:6500c
  ruiser
- 4e5bf89d51171b557ffe5d7fe608037064d6274e28b74aca23cf1f43d4069b08:82001
  ighting
- 51ae9d43482b81a8625f3a31aa3e047f4f06f13294b3d38f8937df46c616b587:06001
  ove00
- c9e247812171f591f0566c5e22f7b873aa43780e53d4e5152842543559a0b753:6400p
  urple!
- 5ed6196959fde9a53bd6ccad3f2acbb0c6983e2b956d9fb50d6d1ca4188156d3:26010
  2031a
- c55a2b0f02f83472dc2d037d7e35b172ae10901c3330d4266bedd6b9fffb8b52:73010
  886k
- 2a5ce6de31ffc39bdd67e419e999b70029a5043da9bbc3ed8d29c048a252d65b:66012
  wrestbg
- edd958c8f98af7a2a6529a2640fc73d660f73ffe76e8787cbf2c0a6d218cd967:66014
  7asdf
- 9e8ab479a3d9df3fe252a42ba1dd8b7c1593bde0afead27d123c6c153a222977:13016
  1563;
- [s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit => q
-
-
- Session.Name....: hashcat
- Status.....: Aborted
- Input.Left.....: Mask (?d?d) [2]
- Input.Right....: File (../.../dict1.txt)
- Hash.Target....: File (formspring.txt)
- Hash.Type.....: SHA256
- Time.Started...: Thu Feb  9 03:07:59 2017 (1 sec)
- Time.Estimated.: Thu Feb  9 03:11:58 2017 (3 mins, 55 secs)
- Speed.Dev.#1...: 6679.0 kH/s (12.52ms)
- Recovered.....: 53/419564 (0.01%) Digests, 0/1 (0.00%) Salts
- Recovered/Time.: CUR:N/A,N/A,N/A AVG:1733.84,104030.41,2496730.00
  (Min,Hour,Day)
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
- Progress.....: 12219200/1584421800 (0.77%)
- Rejected.....: 169600/12219200 (1.39%)
- Restore.Point...: 122191/15844218 (0.77%)
-
- Started: Thu Feb 9 03:07:59 2017
- Stopped: Thu Feb 9 03:08:04 2017
- root@kali:~/Desktop/PasswordCrack/isp/Password List/Password List#
```

● HASH CAT – RULE ATTACK – Rule Online and Custom Rule

- Used Both Rule available Online and a Custom Rule
- Rule Online – Hashcat Blog – Obtained a rule from Hashcat Blog online at location – <https://hashcat.net/forum/thread-4580.html>
- Rule Attack uncovered some valuable amount of passwords

Output : Aborted in few seconds to record the output

```
root@kali:~/Desktop/PasswordCrack/isp/Password List/Password List# hashcat -m
1400 --potfile-disable formspring.txt ../../dict1.txt -r
../../rule1.rule
hashcat (v3.10) starting...
```

OpenCL Platform #1: Intel(R) Corporation

=====

- Device #1: Intel(R) Core(TM) i5-6360U CPU @ 2.00GHz, 501/2004 MB
allocatable, 2MCU

OpenCL Platform #2: Mesa, skipped! No OpenCL compatible devices found

```
Hashes: 419564 hashes; 419564 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 5000
Applicable Optimizers:
* Zero-Byte
* Precompute-Init
* Precompute-Merkle-Demgard
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Salt
* Raw-Hash
Watchdog: Temperature abort trigger disabled
Watchdog: Temperature retain trigger disabled
```

Cache-hit dictionary stats ../../dict1.txt: 193576069 bytes, 15844218
words, 79221090000 keypace

```
fa583069b998c7586bfe9662d8c62bac6438c735ab3f98359da046dea943be30:73happy!!
ee916bdebb40d800923f4a09701e2e3ea8ab14e6aca317016b87dcb4b6fffa27:88louise!!
f4b5529e53a10245de505af6cf2931bb144f876617fbd7b4d0a3661847eb67da:32rain!!
57f76df0c45a161bf9eb1e591518e3359a5d73425979fa293f2f29ef4c4c6821:98!!password
49096c2552120814aa09704baa97430ea7e616df18ab5f2ad91b0b61d875fc4e:62!1kimchi
60cae5cdde7cc36b6e95c835383db8d047b248355ed3004821e50d74e2205603:62!@#QWEASD
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
108f6ae5888c4cf1899b8559056ef98666606bb427745d97ed7fdd654224aaale:08!hawthorne
!1
ccea52791a7f5394ce193641785b67721008699e1d30a4396a66d7cb11550b15:96!iloveyou!
d85fb44692880b9d1716e6e7187b2eb323ca39b2ab74368e26100409e9e4daf9:78elaniel
f83efc3420972bc6c52db93ccc581d78d383554241041ddb8fd467844cb21261:50!!pokemon
1134362d3c422d03b420d64db3124aeb52c6d89672e29604aefeffc5609ca1:46!retep
fb325ad4fd7861afae58eeaf17d8403c383778980ccc538bed884198225ba6fe:65miley!
aeb13ce6ad9a98381537ffde4d3fabcd3b8eda6d6fabce01ce2ffcb42dbe3448:38miley!
c6645085e4ce9bfd250388bb077b610ff98848ecb397255cea946fd2e2ffa960:79"faizan786
"
d4b63e091f22d55b15cbf2915585589496cd7888d2eb872497dd696d70b8b6ca:98##eric$$
6298f33d5f1316303cd7899e9c32c2554192af046a070b9b0de37b42c720eac3:27blink182gi
rl
c48e94c672d0e767ac82a08ab66fc9cbfa825ac6b50a81b8d35c1bbad3b6e90a:76#1biker
8049cc1b77312888c27cdb2b43673e33bc53293ce27e55d2cb3af25cc7d49d22:73#1chick
3113ea8be7fd50fd9bc7688e92354682aebbd2a2bfb19e7b9e9b2cbdeaac8ed2:16#1chick
2308f38415e9bbea5f077c85bf2bc4147d0d6f732e8172365f14060f280850b3:01#1singer
93bda05dca68a6f696e6b0a80bad92e77db28d7cb33a2d6944fb125f431f3945:86#1shopper
0fd405f18e3123a99c8be6e85709c351441d9bead3706496a83b6ecbd9550c6d:83#1singer
d51161fe7800e417dbc97e1628ea152f1de92f50a707bcd9d56a7fcba3ac57a5:48#2funny
4e4a117d504c13678eadf3bca2934c65e18cc22c500ac5b718a16ad8efb4e22f:35#1preston
[s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit => q
```

```
Session.Name....: hashcat
Status.....: Aborted
Rules.Type.....: File (../../../../rule1.rule)
Input.Mode.....: File (../../../../dict1.txt)
Hash.Target....: File (formspring.txt)
Hash.Type.....: SHA256
Time.Started...: Thu Feb  9 03:11:25 2017 (8 secs)
Time.Estimated.: Thu Feb  9 05:36:10 2017 (2 hours, 24 mins)
Speed.Dev.#1...: 9122.8 kH/s (11.58ms)
Recovered.....: 24/419564 (0.01%) Digests, 0/1 (0.00%) Salts
Recovered/Time.: CUR:N/A,N/A,N/A AVG:171.66,10299.45,247186.78 (Min,Hour,Day)
Progress.....: 81858504/79221090000 (0.10%)
Rejected.....: 2765000/81858504 (3.38%)
Restore.Point...: 15981/15844218 (0.10%)
```

Started: Thu Feb 9 03:11:25 2017

Stopped: Thu Feb 9 03:11:37 2017

root@kali:~/Desktop/PasswordCrack/isp/Password List/Password List#

```
- SHA1 with Custom rules - Custom Rule --- hashcat -m 100 <Password
Hash> <Dictionary> -r <Custo Rule>
Rules condition: ^00l
                  ^00u
                  ^00
                  ^99l
                  ^99u
                  ^99
                  $00
```

Truncating the beginning and end with 2 digits (for full upper case full lower case and as from the wordlist), similarly extended rules for single digit and three digit patterns

Srinivas Piskala Ganesh Babu (N13138339) and spg349

- Tried to use the GPU for processing the Hashing and match code there, Have issues with the drivers of nvidia. Hoping to solve it !

- The XSplit File consisted of a lot of data with the hashes, in the format Username Email and the Hash. The First task is to extract the hashes
- The file consisted of hashed password – hashed with sha1
- **1. Extract the hashes into a new file: Python and Regular Expression**

- This Program outputs a file which will contain only the hashes in a file
- Identified and analyzed the hash type by brute forcing from SHA1 using hashcat – Dictionary attack – Cracked with SHA1 itself
- The file consisted of hashed password – hashed with sha1
- Identified the Hashes by the structure of the hash

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

- Tried RAW SHA1 Option with flag (-m 100) in HASHCAT and it proved to hit some.
- Methods similar to the Linkedin Scenario

- **TECHNIQUE USED:**

- **HASH CAT – RAW SHA1 HASH Dictionary Attack**

- Used SHA1 option to perform a dictionary attack
- **Output : Aborted to record the output**
- `root@kali: ~/Desktop/PasswordCrack/isp/Password List/Password List#`
`hashcat -m 100 --potfile-disable ../../xsplint.txt ../../../dict1.txt`
- hashcat (v3.10) starting...
-
- OpenCL Platform #1: Intel(R) Corporation
- =====
- - Device #1: Intel(R) Core(TM) i5-6360U CPU @ 2.00GHz, 501/2004 MB
allocatable, 2MCU
-
- OpenCL Platform #2: Mesa, skipped! No OpenCL compatible devices found
-
- Hashes: 2499789 hashes; 1919651 unique digests, 1 unique salts
- Bitmaps: 18 bits, 262144 entries, 0x0003ffff mask, 1048576 bytes, 5/13
rotates
- Rules: 1
- Applicable Optimizers:
- * Zero-Byte
- * Precompute-Init
- * Precompute-Merkle-Demgard
- * Early-Skip
- * Not-Salted
- * Not-Iterated
- * Single-Salt
- * Raw-Hash
- Watchdog: Temperature abort trigger disabled
- Watchdog: Temperature retain trigger disabled
-
- da39a3ee5e6b4b0d3255bfef95601890afd80709:
- [s]tatus [p]ause [r]esume [b]ypass [c]heckpoint [q]uit => Cache-hit
dictionary stats ../../../dict1.txt: 193576069 bytes, 15844218
words, 15844218 keypace
-
- 742ce30a73b59259a9b55e5eaf0e97e813167d60:-----
- 59c52c1ffde06261d5ef60e76b2b5124d72a109b:-----
- 965235309ce4cd648462f1d6d27ba7ff9e6e9019:--121212r
- 8d43bae4454b598c89520716b2736d2e5d7bba1d:-1 OR 1=1
- 7751de20fefdbab5b49ff18c875ccc0f6ef3a6cd8:-198989t
- 0cabd13b78b201536eeff866622530b94d34dde4:-1a2b3c-

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

```
- 21aa01bcd657bacb0b701a5c5a239c81eb552d87:-50ee2f11-
- 6a7ef56002e95174993d5e419ac05a385d2221c1:-Tonil980
- b9e9c0c203f110f4fb9242df98bb68e2bb835d09:-a4490a-
- bc6792151ee6d742aac780380ad7a43f6eec763f:-arminia
- bfec557479e9a0fa3f3369542e9448296314ae90:-compnad1
- 7ea91a076d17ca022a54d44e64f63281506c04e4:-maus-
- 5bd9f416f264ab10895ec88de2a453fe18elec09:-nieheu-
- f83cf4304bbf3db773b1688178a06eaa81d0135a:-p0o9i8u
- aa7d3d1e4b687b78e85575708c56db09a90f39b3:-p0o9i8u7y
- 8d2e5705f0bc3fcd91a894cbdc0cb16c9db6ae1a:-pl0ok9ij
- 7025f4aedb8edefd95b86d480f9169ecfff2b40e:-yoshi91
- cd5dac8d8ec9dc53061bde313bc29260b3be536b:...---...
- 93fcbf4c3221f6e6fc21e07ec6e87ad94e60bd26:.....
- c77c9523a46ead17e1332542ed0db5867130dlb7:.....
- f0elcal6e77dd71ec18f2a537948c27c5317304a:.....
- 679509dc972f75700f18e083a71a1c59283f84a6:.....
- babd0c08a681c021012bed05be9e51ca617e2b04:.....
- falc364c3bc40b58335a02bfabbfba6ab37d9b76:....1234
- d422b84f530d47aeefe84032335218885f2881ce5:...123
- 374700572bd9c376bf6fdf30e9a6cf884d254722:...damian543..
...
- ddf6a104c328ee414cbc62259181d3e0f5a3337d:.adgjm
- ba8fadaf31d5e9e95f85a47dbb4172447c187c15:0258963
- b68ced2edb527f8cc136374fedfc8aaf107e7b4c:02593990
- 42a4d2e613abf803a7fcf2647b807ab90ec98819:026026
- 054968b3c8afdf0e4f665db50b82495a2a84bfca:02610261
- e33d0871cf7ab89c62a605dff82a88420d05cb41:026159710
- 9d923a8959354270958000d5c0e830076ee87f87:026326
- d25e18464f478c132eed375c229404f5b9447ff6:0263541zxc
- 4ddaf2c2e75403768c11f46e6c0e373c2ba4eb48:030681
- 624e64027d30f410253e6545f71fd68bbd311db4:030683
-
- Session.Name....: hashcat
- Status.....: Aborted
- Input.Mode.....: File (../ ../dict1.txt)
- Hash.Target.....: File (../ ../xsplitt.txt)
- Hash.Type.....: SHA1
- Time.Started....: 0 secs
- Time.Estimated.: Thu Feb 9 03:25:29 2017 (56 secs)
- Speed.Dev.#1....: 277.0 kH/s (1.35ms)
- Recovered.....: 3220/1919651 (0.17%) Digests, 0/1 (0.00%) Salts
- Recovered/Time.: CUR:N/A,N/A,N/A
  AVG:300222.44,18013346.00,432320320.00 (Min,Hour,Day)
- Progress.....: 172078/15844218 (1.09%)
- Rejected.....: 2094/172078 (1.22%)
- Restore.Point...: 170029/15844218 (1.07%)
-
- Started: Thu Feb 9 03:24:25 2017
- Stopped: Thu Feb 9 03:24:34 2017
- root@kali:~/Desktop/PasswordCrack/isp/Password List/Password List#
```

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

- **Python Script with Hashlib Module:**

The script basically uses hashlib module of python

- Reads the Hashes
- Made Wordlist initially, using the Username and Email given in thexsplit_leak sheet
- Reads the Dictionary for wordlist
- Performs SHA1 Digest to the wordlist one by one
- Compare the hashed(wordlist) with the LinkedIn Hash
- Hit up if there is a match else continue – Write Matches to a file
- Python script used is found below

```
- import hashlib,sys
def main():
    print " SHA1 Password Hashing Crack"
    print " XSplit: Hash Type - SHA1 - SHA1(Password)"

    fd = open("/home/srinivas/Desktop/PasswordCrack/Password List/XSplit.txt", "r")
    hashes = fd.readlines()
    fd.close()

    fd = open("/home/srinivas/Desktop/PasswordCrack/test1.txt", "r")
    data = fd.readlines()
    fd.close()
    count = 0

    ans = open("/home/srinivas/Desktop/PasswordCrack/answers.txt", "a")

    try:
        for hash in hashes:
            for dic in data:
                h = hashlib.sha1(dic.strip("\r\n")).hexdigest()
                if h == hash.strip("\r\n"):
                    print "Password Hit for %s\n" % (dic)
                    ans.write("%s %s\n" % (hash.strip(),dic.strip()))
                    count = count + 1

    except:
        print "Test Done !!! Count = %d" % (count)
        print "Test Done !!! Count = %d" % (count)

main()
```

- Output Below: Stopped after certain hits

```
/usr/bin/python2.7 /home/srinivas/PycharmProjects/PasswordCrack/linkedin.py
SHA1 Password Hashing Crack
XSplit: Hash Type - SHA1 - SHA1(Password)
Password Hit for moralan
```


Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

Password Hit for a4ndre3

Password Hit for intelinside

Password Hit for thomas42

Password Hit for fucknut45

Password Hit for bonnie97

Password Hit for ancient1

Password Hit for 600304

Password Hit for ace123

Test Done !!! Count = 9

Process finished with exit code 0

- **Online Decryptor – HashKiller (hashkiller.co.uk) –**
Used an online hash decryptor like the one mentioned hashkiller

```

We found 34 hashes! (Timer: 285 m/s) Please find them below...

aeb8d80cae53ccb37cb6e189ef455ecdd2dabdea SHA1 : Masterdl6
65a592ce6513e8ff06a283b06eaa43c7c000227 SHA1 : toraks518
e5121cd715956e6a8f517d2cc1eae0003f1ba0c SHA1 : cenation11
9ef1c18e4d0c3b5c8e4eb3e5010a58d3578c765 SHA1 : alimahady78
6856c2dbf9a42176d640e1a93c9bc93204986118 SHA1 : blackberrie5
d60025ad9fa2a523c9a3f945c2bba5f32c1e0 SHA1 : stven546
da037bcb4e4c9f6a2b0c808f5181b3e0cd4ade SHA1 : aktra629
17e7b09a2144dc51ac2a97796081bc67d8a3dcf4 SHA1 : maxwell11
d620b9c26a395081a0dd335d3b94ecb9b7f4ec8 SHA1 : bearsrule
a3e3cb40f7ec229e70d5db47072d8d37ea18c7 SHA1 : MyPeanutword
276ca25e858ef1dd4f712c9eb56aaa56208ac82 SHA1 : stowie4289
f6409e21bd7ed05d35f5ae163a5eb4690dd93245 SHA1 : sangodayo
9cb78341457257304a979c658ad8689f487c744 SHA1 : Raitlm2master
26433897fca67002cb8edd5d93a4db34e79ab04 SHA1 : santasale
56309d38411cd0f7b0d7e3eeaaa4bda8a5f60 SHA1 : robbiel23
fe44c0bbb85257d88feb04b841d9df5f6191ba4 SHA1 : salocin--64
3bda72d3f7247116654d480220907708eae9bc4 SHA1 : f89tk1a02
db252fcd4cd2d2b1e7a307241f548e03c312a SHA1 : qee1234
6ad857c2a9b8f6c23f9d550a73c0375b057f07c7 SHA1 : r5vq7777
d59a26ccfd74040a1fd612e725019f5f8a1d7970 SHA1 : zd88a77g
e01ae47f0fc06b5b49cd6b38b9deedcma432499 SHA1 : mhertg2ab
1902f00433efrdbca3f9e77e83888d10a1ac362 SHA1 : ruby2004
bfc2a7a16b73861fee2079b0ee27690a3beaffae SHA1 : 01hW19
50388acc2489503e852530f4e6edc0eb640347a SHA1 : jwdawg11
7282f85f4e5a5828d8d0e0d412b3307c254 SHA1 : jank9777
79987fbc951a0c889aefcc15ab37257dc0c9948 SHA1 : aedandl34
c283a9fa080cc45dfae20598412074d759d2fff3 SHA1 : sironumal467
ef432d89c89c1698d6f638d5f847593f12f18b1d SHA1 : taro1204
34ae71744574ed5cb92155e48d6bcb2b09e98 SHA1 : clessclass
62fe73a8de05e7f42c642b2af42b0d12483bbd SHA1 : sanngp
a8999e4acbf7e826b532f0fd3bde962fe6023c SHA1 : armadylgodaword
5aa9750512cac30e3b8d0a7cb1b03d368fc2e999 SHA1 : alexander56
f31e7979ad4cd4f33b0d28cc9e94baf3b0cf0 SHA1 : slfjgnty97
08a64d41b24d4d2fe40bd16fe74cca207e7b63ad SHA1 : osiris-teign
```

- **HASHCAT – Rule and Hybrid Attack with Different Dictionaries**
Peformed a combination attack using the Using the corresponding flags
---- Rule + Dictionary (-r Rule + Dict)
---- Mask + Dictionary (-a 7 ?d?d Dict or -a 7 Dict ?d?d)

Internet Security and Privacy Assignment – 2

Srinivas Piskala Ganesh Babu (N13138339) and spg349

Updated the results in the file attached for the cracked lists combining all the above results and attached with the zip file