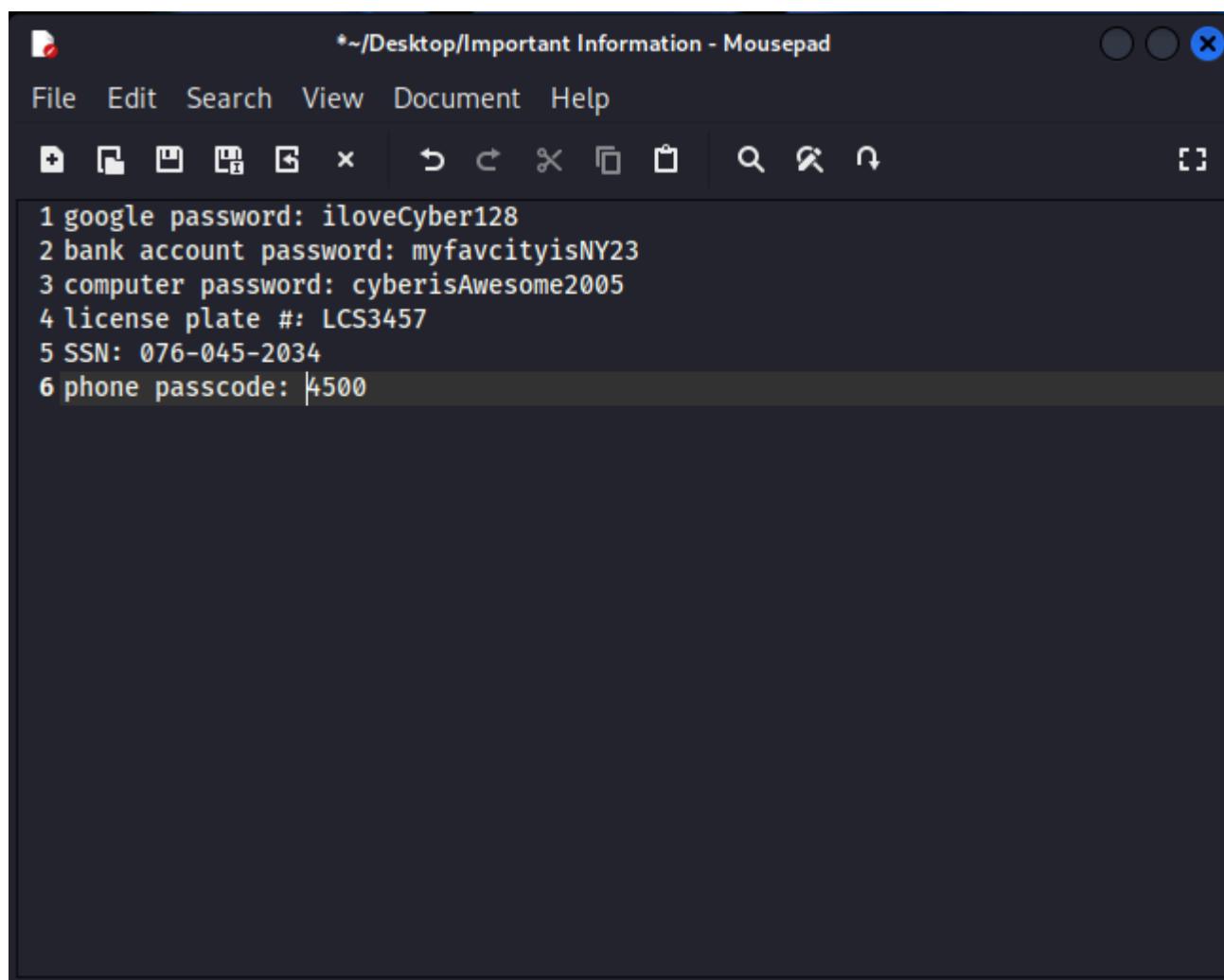


The purpose of this lab is to practice different attacks on different components of a machine. I will be using a virtual machine running Kali Linux to create and execute these exploits.

I will create password-encrypted files and use John the Ripper to crack these passwords. The goal is to identify which passwords were considered strong and which were weak.

These are the files I want to password-encrypt because they have sensitive data that I want to protect.



The screenshot shows a terminal window titled "Mousepad" with the file path "/Desktop/Important Information". The window contains a list of sensitive information items, each preceded by a number from 1 to 6:

- 1 google password: iloveCyber128
- 2 bank account password: myfavcityisNY23
- 3 computer password: cyberisAwesome2005
- 4 license plate #: LCS3457
- 5 SSN: 076-045-2034
- 6 phone passcode: 4500

The screenshot shows a terminal window titled '*~/Desktop/Client List - Mousepad'. The window has a dark theme and contains the following text:

```
1 Client List:  
2  
3 Client 1: Ashley White  
4 Address: 45 Smith Street, Oklahoma City, OK 7308  
5 Phone Number: 572-098-1224  
6 Spouse: Cam White  
7  
8  
9 Client 2: Amy Booker  
10 Address 87 Cream Ave, Miami, Fl 33101  
11 Phone Number: 305-477-1278  
12 Spouse: N/A
```

Now I will encrypt both of these files with passwords using the GPG command. One with a strong password and one with a weaker password.





GPG is one of the strongest password encrypters and simply using the gpg command again can reverse the encryption, but for this experiment I want to use John The Ripper to actually find out what password was used to encrypt.

Even with a relatively strong password (the weaker password I used), according to the updates I get when I extracted the hash and attempt to break it, it takes approximately 1 hour. .

```
(kali㉿kali)-[~]
$ cd Desktop
(kali㉿kali)-[~/Desktop]
$ gpg2john 'Client List.gpg' > gpphash.txt
File Client List.gpg
(kali㉿kali)-[~/Desktop]
$ john gpphash.txt
stat: gpphash.txt: No such file or directory
(kali㉿kali)-[~/Desktop]
$ john gpphash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (gpg, OpenPGP / GnuPG Secret Key [32/64])
Cost 1 (szk-count) is 65011712 for all loaded hashes
Cost 2 (hash algorithm [1:MD5 2:SHA1 3:RIPEMD160 8:SHA256 9:SHA384 10:SHA512 11:SHA224]) is 10 for all loaded hashes
Cost 3 (cipher algorithm [1:IDEA 2:3DES 3:CAST5 4:Blowfish 7:AES128 8:AES192 9: AES256 10:Twofish 11:Camellia128 12:Camellia192 13:Camellia256]) is 9 for all loaded hashes
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
og 0:00:00:13 0.29% 2/3 (ETA: 19:02:09) og/s 38.34p/s 38.34c/s 38.34C/s chloe..compaq
og 0:00:01:24 1.62% 2/3 (ETA: 19:13:59) og/s 38.19p/s 38.19c/s 38.19C/s trash..turbo2
og 0:00:01:59 3.98% 2/3 (ETA: 18:37:11) og/s 38.19p/s 38.19c/s 38.19C/s Gemini..Germany
og 0:00:02:00 4.00% 2/3 (ETA: 18:37:23) og/s 38.19p/s 38.19c/s 38.19C/s Jaguar..Jenny1
og 0:00:02:49 5.02% 2/3 (ETA: 18:43:28) og/s 38.14p/s 38.14c/s 38.14C/s Gilgamesh..Goethe
og 0:00:02:50 5.04% 2/3 (ETA: 18:43:36) og/s 38.15p/s 38.15c/s 38.15C/s Jettai..Joy
og 0:00:04:32 7.25% 2/3 (ETA: 18:49:50) og/s 38.08p/s 38.08c/s 38.08C/s minnie1..mouse1
og 0:00:05:07 7.98% 2/3 (ETA: 18:51:28) og/s 38.09p/s 38.09c/s 38.09C/s vanessa1..vermont1
og 0:00:05:08 8.01% 2/3 (ETA: 18:51:27) og/s 38.09p/s 38.09c/s 38.09C/s action1..artist1
```

With the stronger password it takes twice as long and I had to create a new session to ensure that I could crack both at the same time. I will let them run and see what happens.

```
(kali㉿kali)-[~/Desktop]
$ gpg2john 'Important Information.gpg' > file2_hash.txt

File Important Information.gpg

(kali㉿kali)-[~/Desktop]
$ john file2_hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (gpg, OpenPGP / GnuPG Secret Key [32/64])
Cost 1 (s2k-count) is 65011712 for all loaded hashes
Cost 2 (hash algorithm [1:MD5 2:SHA1 3:RIPEMD160 8:SHA256 9:SHA384 10:SHA512
11:SHA224]) is 10 for all loaded hashes
Cost 3 (cipher algorithm [1:IDEA 2:3DES 3:CAST5 4:Blowfish 7:AES128 8: AES192
9: AES256 10:Twofish 11:Camellia128 12:Camellia192 13:Camellia256]) is 9 for a
ll loaded hashes
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Crash recovery file is locked: /home/kali/.john/john.rec

(kali㉿kali)-[~/Desktop]
$ john --session=gpg_file2 file2_hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (gpg, OpenPGP / GnuPG Secret Key [32/64])
Cost 1 (s2k-count) is 65011712 for all loaded hashes
Cost 2 (hash algorithm [1:MD5 2:SHA1 3:RIPEMD160 8:SHA256 9:SHA384 10:SHA512
11:SHA224]) is 10 for all loaded hashes
Cost 3 (cipher algorithm [1:IDEA 2:3DES 3:CAST5 4:Blowfish 7:AES128 8: AES192
9: AES256 10:Twofish 11:Camellia128 12:Camellia192 13:Camellia256]) is 9 for a
ll loaded hashes
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
0g 0:00:00:11 0.15% 2/3 (ETA: 20:11:17) 0g/s 18.91p/s 18.91c/s sophi
e..stephen
0g 0:00:00:53 0.53% 2/3 (ETA: 20:59:15) 0g/s 18.53p/s 18.53c/s 18.53C/s nirvana1..notebook
0g 0:00:00:55 0.54% 2/3 (ETA: 21:00:04) 0g/s 18.51p/s 18.51c/s 18.51C/s queenie ..random
```

Unfortunately, I could not crack these passwords in time for this lab, but my research does show that strong passwords made using GPG are intended to take time to crack and sometimes can be infeasible. In this case, if this password cracking were time sensitive, this would be infeasible.