ACIT 4630 – Lab 7 – Password Cracking

Notes:

This lab should be done with your partner (if you have one). Provide screenshots of the commands you run and the results.

Instructions:

Important Note: If running Windows, please make sure Hyper-V, Virtual Machine Platform, and Windows Hypervisor Platform are turned off in Windows Feature on your laptop.

Inside Kali, open a terminal and elevate to a root shell (or, alternatively, run all commands below with sudo): `sudo -s`

Part 1: Online Password Attack

In this section, you will launch attacks over the network against login services, to brute-force a username/password.

This attack uses all unique passwords from the 32 million RockYou password breach. If the file `/usr/share/wordlists/rockyou.txt` is not present on the VM, unzip the password list.

```
gunzip /usr/share/wordlists/rockyou.txt.gz
```

You are going to use a tool called Hydra (already installed in Kali Linux) to use this password list and launch an online password attack against a website that is managed by Dr. Eargle. Dr. Eargle **authorizes** you to launch the attack only as specified in the instructions below:

- Let's say that you wanted to crack the password for this route: https://is.theorizeit.org/auth/
- Imagine that you knew, or guessed, that one of the usernames was `istheory`.

Show that you're able to login to the website successfully. Hint use `hydra -h` to see some examples.

Answer these questions:

- **Q1.** Why is this attack considered an online attack?
- **Q2.** What kind of password attack did we do in this task?
- **Q3.** List three countermeasures that organizations could implement specifically against online attacks.
- **Q4.** Explain if online attacks are feasible choices for an **exhaustive** (trying all different combinations) brute-force attack.

Side Note: In Greek and Roman mythology, Hyrda is a mythical sea monster with many heads. When a head is cut off, it is replaced by another. By default, the tool Hyrda runs with 16 concurrent heads (tasks), each of which makes a battery of attacks (password guesses) before dying off and being replaced by another head (task).

Part 2: Offline Attack Using Hashcat

While in the previous section attacks were launched over the network, in this section, you will start with a *password hash* stored on your computer, and you will launch attacks against it locally. No authentication service such as a web server sits in front of your attempts, potentially throttling the speed at which you can make guesses. Instead, you can make guesses as fast as your computer and the hash algorithm will allow.

First, a dictionary attack will be performed against a password-protected Word document. We will use a tool called Hashcat (already installed in Kali Linux).

1. Run the following command to get the password-protected Word document:

   ```
   wget https://raw.githubusercontent.com/deargle/security-assignments/master/labs/files/hashcat.doc
   ```

2. Now run the following command to get a Python script that extracts the hash of a password-protected Word document. (The hash of the password is stored within the metadata of the document file itself.) `john` in that script's name refers to JtR, John the Ripper

   ```
   wget https://raw.githubusercontent.com/magnumripper/JohnTheRipper/bleeding-jumbo/run/office2john.py
   ```

3. Next, run the following command from the directory to where you downloaded `office2john.py` and `hashcat.doc`:

   ```
   python office2john.py hashcat.doc
   ```

4. We need to convert the output to a format that hashcat can recognize. To do so, copy from the first $ until but not including :::

   For example, if the output is

   ```
   hashcat.doc:$oldoffice$1*04477077758555626246182730342136*b1b72ff351e41a7c68f6b45:::hashcat.doc
   ```

   you need to extract:

   ```
   $oldoffice$1*04477077758555626246182730342136*b1b72ff351e41a7c68f6b45
   ```

   Save the extracted hash into a file in your home directory. Name the file whatever you like.

   Note: make sure the entire hash is on one line within the text file. **Don't add extra spaces at the end.** If you get a "line-length exception" in the next step, make sure there's not a typo at the beginning of the hash.

5. Run the following command to launch a dictionary attack to crack the hashed password (Remove the <>, use whatever name you want for the output file):

```
hashcat --force -a 0 -m 9700  -o <outputFileName.txt>
<HashInputFileName.txt> /usr/share/wordlists/rockyou.txt
```

Answer these questions:

- **Q5.** What is the password for hashcat.doc? (open the output file and look at the end of the hash)
  - Download the word file on your computer and verify you can see the content.
- **Q6.** What kind of password attack did we do in this task?
- **Q7.** How is this offline password attack different from the online hydra attack you attempted earlier?
- **Q8.** List three countermeasures that organizations could implement specifically against offline attacks.

Part 3: Cracking LinkedIn Hashes Using Hashcat

In this section, you'll see how many hashes you can recover from the 2016 LinkedIn password breach. The LinkedIn hacker, a Russian, was sentenced in US court to seven years in jail on September 29, 2020. This breach of 177,500,189 **unsalted SHA1** password hashes represents the data of all LinkedIn users as of 2012. Among these passwords, only 61,829,207 are unique.

1. Download a copy of 500,000 of these passwords:

```
wget https://raw.githubusercontent.com/deargle/security-
assignments/master/labs/files/LinkedIn_HalfMillionHashes.txt
```

2. Run the following command to perform a "straight" dictionary attack using the rockyou.txt wordlist again:

```
hashcat --force -m 100 --remove --outfile=LinkedIn_cracked.txt
LinkedIn_HalfMillionHashes.txt /usr/share/wordlists/rockyou.txt
```

   - **Note:** The above command may take a couple of minutes to run. To see the status of a running job in Hashcat, press the `s` key (it might take up to 15 seconds for Hashcat to report its status).
   - **Note:** Notice the `--remove` flag. This will remove cracked hashes from the input file. So, if you run these commands more than once without changing anything, it won't crack anything after the first time.
   - **Note:** You can always open another terminal session and count the number of lines in your outfile to see how many you've cracked so far, in total:

```
wc -l LinkedIn_cracked.txt
```

Or count the number of passwords left (it started with half a million):

```
wc -l LinkedIn_HalfMillionHashes.txt
```

To see hashes cracked in real-time, in another terminal shell, type the command (Use Ctrl+C to exit the tail command):

```
tail -f LinkedIn_cracked.txt
```

- **Note:** If you accidentally delete your cracked outfile, you will need to delete your hashcat "potfile" too before you try to recreate the outfile. You need to do this because otherwise, hashcat won't write any already cracked hashes found in the potfile to the outfile. The `hashcat.potfile` is stored in a hidden direction in the home directory of whomever you run the command as.

  To do this run `rm ~/.hashcat/hashcat.potfile`

  Don't forget to also start with a fresh 500k hashlist, because the `--remove` flag would have removed rows from that file as the hashes were cracked and inserted into the potfile.

- **Q9.** What's the purpose of `-m 100` option in the above command? (Hint: `man hashcat`)
- **Q10.** What kind of password attack did we do in this task?
- **Q11.** How many passwords were you able to recover using the Hashcat command above?

2. Now run another attack that uses a rules-based method. Rules apply common patterns to password dictionaries to crack even more hashes. You can read about rules in Hashcat here: https://hashcat.net/wiki/doku.php?id=rule_based_attack.

   The “best64.rule” is one of the most effective sets of Hashcat rules.

   ```
   hashcat --force -m 100 --remove --outfile=LinkedIn_cracked.txt
   LinkedIn_HalfMillionHashes.txt -r /usr/share/hashcat/rules/best64.rule
   /usr/share/wordlists/rockyou.txt
   ```

   - **Q12.** What is the purpose of using rules in this command?
   - **Q13.** How many total passwords were you able to recover after using this rules-based attack in combination with the earlier straight attack?
   - **Optional**: Experiment with other rules found in: `/usr/share/hashcast/rules`
3. Run another attack that uses a hybrid method using a dictionary attack combined with a “mask,” which is a pattern that is appended to each password in the password dictionary:

```
hashcat --force -m 100  --remove --outfile=LinkedIn_cracked.txt
LinkedIn_HalfMillionHashes.txt -i -a 6
/usr/share/wordlists/rockyou.txt ?d?d
```

The `?d?d` at the end means to append two digits between 0–9 each at the end of each password in the rockyou.txt password dictionary.

- **Note:** `-i` (short for `--increment`) iterates through the given mask until the full length is reached. So in the above command, both masks of `?d` and `?d?d` will be used.
- **Q14.** What does option `-a 6` mean in the above command?
- **Q15.** How many total passwords were you able to recover after using this hybrid attack combined with the earlier straight and rule-based attacks?
- **Optional:** Try using a different character set for your mask. Note that each mask below is for one character. If you wanted to test four digits at the end of each password, the mask would be: `?d?d?d?d`

```
?l = abcdefghijklmnopqrstuvwxyz
?u = ABCDEFGHIJKLMNOPQRSTUVWXYZ
?d = 0123456789
?s =  !"#$%&amp;'()*+,-./:;&lt;=&gt;?@[\]^_`{|}~
?a = ?l?u?d?s
?b = 0x00 -  0xff
```

4. Another common password pattern is to prepend digits at the beginning of passwords. Try this mask, by running the following command:

```
hashcat --force -m 100  --remove --outfile=LinkedIn_cracked.txt
LinkedIn_HalfMillionHashes.txt -i -a 7 ?d?d
/usr/share/wordlists/rockyou.txt
```

- **Note:** Notice `-a 7` option in the above command
- **Q16.** How many total passwords were you able to recover after using this hybrid attack combined with all earlier attacks?

<u>Submission For Lab 7:</u>

- Create a report answering any questions in the lab above and including any required screenshots.
- Walk through your report with your instructor.
- Submit your report to the Learning Hub in PDF format.

This lab is due at the end of the class and is worth a maximum of 10 marks.