Project 3 – Password Cracking, Computer and Network Security, CSE 467/567
Department of Computer Science and Engineering, Miami University

Objectives

The purpose of this lab is to learn how to use a basic password cracker: **John the Ripper (JTR)** to recover passwords from stored password hashes. JTR is a popular program for cracking passwords (It is available for free from www.openwall.com/john if you would like a copy on your own computer). Before using it, you must make a copy of its configuration files, which must be present in the directory you are running the program from.

Background

- Please review the slides for password cracking.
- We will be using the Kali virtual machine for this lab, which already has **John the Ripper** installed on it.
- JTR rules: https://www.openwall.com/john/doc/RULES.shtml
- JTR examples: https://www.openwall.com/john/doc/EXAMPLES.shtml
- Common password lists: examples here: https://www.openwall.com/wordlists/

Tasks:

Create a directory inside the "LABS" directory called "password-cracking" (`~/LABS/password-cracking`). **Do all your work for this lab in this directory, unless otherwise specified.** Please copy the attached zip file to the password-cracking directory and unzip it. (*hint:* use scp and unzip commands)

1. (10 points) Take a look at the `/etc/shadow` file.
    a. When passwords are stored in `/etc/shadow`, they usually follow the format `username:id:salt:hash:lastPasswordChange::::::`. The `id` is the algorithm GNU/Linux uses to generate the hash value for a given password (i.e. SHA512, MD5, etc…).
    b. The table below lists some different types of `id`'s. Do some research online and find their associated hashing algorithm. I've provided one answer already. **Record your answers in the table**.

ID	Hash Algorithm
*	Not hashable, user can't login
!	Not hashable, user can't login
1	**MD5**
$2a$	Blowfish
$2y$	Eksblowfish
5	SHA256
6	SHA512

All activities practiced in this course are done in a controlled laboratory environment. Doing any of these activities on computers for which you do not have permission **CAN BE CONSIDERED A CYBERCRIME** and you would face legal consequence.

y	Yescrypt

2. (10 points) Run JTR on the shadow file of your Kali VM and see if it can crack any passwords
    a. Run command `sudo john --nolog --pot="john.pot" --session=john --max-run-time=300 /etc/shadow`
    b. If you could crack it, **please specify the cracked username and password in your lab submission**
        kali:kali
    c. You can use `sudo john --pot="john.pot" --show /etc/shadow` to view `john.pot`, which is where the recovered passwords are stored
    d. Delete `john.pot` (`sudo rm john.pot`) before running the command again JTR again
3. (10 points) Create 5 users in the Kali VM : user1, user2, user3, user4 and user5. Use these passwords respectively: blue, drowssap, abc123, miami20, Ki#di54.
    a. Before you add any users, it is important to understand that Kali has a strong hashing mechanism known as yescrypt by default. When adding a user, Kali will use yesscript to encode their password. JTR will have a hard time breaking these kinds of hash values, so you need to force Kali to use a SHA-512 based hashing algorithm.
    b. Open `/etc/pam.d/common-password` and find the line that that reads "`password [success=1 default=ignore]pam_unix.so obscure yescrypt`"
    c. Change `yescrypt` to `sha512`
    d. Now that you've changed Kali's default hashing algorithm, create the 5 users using `sudo adduser USERNAME`. If you want to change the password for a user, type `sudo passwd USERNAME`.
    e. Run JTR on the shadow file again. Are you able to crack any password? If so, please **list them in your submission**.
        i. **abc123 for user3**
        ii. **blue for user1**
        iii. **drowssap for user2**
        iv. **kali for kali**
4. (10 points) The mode JTR uses for brute force is called "Incremental". JTR stores cracked passwords in a pot file. To run John on the shadow1.txt (supplied in zip file) file, you should run the command: `sudo john --nolog --pot="john.pot" --session=john --incremental=alnum --max-run-time=300 shadow1.txt`. Note that here we force JTR to use alphanumeric characters for a password. How many passwords were cracked in 300 seconds? If all were cracked, how long did this take to crack all the passwords? **Please list the cracked passwords in your submission.**
        i. **abc for user1**
5. (10) If we have prior knowledge of the password format, we can make this process a bit quicker, by using a variation in incremental mode that only checks certain formats.
    a. In the `john.conf` file (by default, `/etc/john/john.conf`), edit the incremental alnum mode to only check passwords of length 1-5.
    b. Now, remove the `john.pot` file and run the command from Step 4. **How do your results differ?**

All activities practiced in this course are done in a controlled laboratory environment. Doing any of these activities on computers for which you do not have permission **CAN BE CONSIDERED A CYBERCRIME** and you would face legal consequence.

   a. **still had abc for user1**
   b. **no new passwords cracked**

c. Run the same command above, using the **shadow2.txt** file instead (and removing the `john.pot` file each time). Are the recovered passwords the same? Did it take less time to recover the passwords this time? **List the cracked passwords and the explanations in your submission.**
   a. **abc for user1**
   b. **mah for user5**
   c. **mdu for user2**

6. (10) Now try running John in incremental mode on **shadow3.txt** file.
   a. Remove the `=alnum` from the command line, since we edited that to only work on short passwords.
   b. Change the runtime to be 10 minutes. This can be done by editing the max-run-time on the command line to say `--max-run-time="600"`. How many passwords was John able to crack in the new file? **List the cracked usernames and passwords.**
      a. **12345 user10**
      b. **secret user20**

7. (10) Obviously, the incremental mode is not so great for more complex passwords. To make some passwords easier, John has a wordlist mode. By default it uses the dictionary in `password.lst`, although other wordlists can be downloaded. Run `sudo john --nolog --pot="john.pot" --session="john" --wordlist shadow3.txt`. **How long does it take and how many passwords are found? Explain the difference between these results and those of the previous question.**
   a. **12345 for user10**
   b. **password for user9**
   c. **password1 for user2**
   d. **qwerty for user3**
   e. **secret for user20**
   f. **ncc1701 for user22**
   g. **Password for user32**
   h. **wordpass for user11**
   i. **a for user36**
   j. **It took about 15 seconds, these results came much quicker because we use the wordlist of common passwords so it is able to check against that instead of randomly guessing**

8. (10) John is also capable of doing simple transformations on the wordlist. This can be done by adding the `--rules` option to the command. **How do the results from this command (time and number cracked) differ from the previous one? Should someone trying to crack passwords always use `--rules` with a wordlist? Explain.**
   a. **12345 for user10**
   b. **password for user9**
   c. **password1 for user2**
   d. **qwerty for user3**
   e. **secret for user20**
   f. **ncc1701 for user22**

g. **Password for user32**
h. **wordpass for user11**
i. **a for user36**
j. **secret1 for user37**
k. **SECRET for user38**
l. **There are 2 more cracked but it took x minutes. You should use –rules if you don't care about time as much in order to crack just a few extra passwords. You should run without –rules first in order to crack the easy passwords quick**

9. (10 points) Use the supplied dictionary as the wordlist (`--wordlist=dic.txt`) to crack the passwords in **shadow4.txt**. Wait about five minutes. **Document how many passwords are cracked and list the passwords.**
   a. **nutmeg for mbear**
   b. **wakeboar for afrien**
   c. **Password for twalle**
   d. **Lightsab for donner**
   e. **Abcd1 for blixen**
   f. **Nana1 for blight**
   g. **Erick1 for eadam**
   h. **Ashin1 for jworld**
   i. **Komodo1 for etwo**
   j. **Square1 for rhide**
   k. **Square1 for riding**
   l. **odnetniN for mbarans**
   m.
10. (10 points) A wealthy family from out of town known as the Rose's just moved to your neighborhood. Their usernames and passwords are stored in `roses.txt`. Run `sudo john --nolog --pot="john.pot" --session="john" --wordlist=adobe_top100.txt roses.txt`. `adobe_top100.txt` contains Adobe's hackers release top 100 most common passwords. **Document how many passwords are cracked and list the usernames/passwords.**
    a. **macromedia for moira**
    b. **monkey for roland**
    c. **princess for alexis**
    d. **football for patrick**
    e. **dreamweaver for twyla**
    f. **555555 for stevie**
    g. **whatever for david**
    h. **freedom for johnny**

Submission:

- Submit a PDF document (prepare with MS Office or Google Doc) with all the required information. For each task, please specify the command you ran. Mention the output as well.
- Upload a single PDF file to canvas assignment. Please prepare the response using standard word processing software such as Google Docs, MS Office, LaTeX.
- Please show each step of your work. **BOLD** the final answer

All activities practiced in this course are done in a controlled laboratory environment. Doing any of these activities on computers for which you do not have permission **CAN BE CONSIDERED A CYBERCRIME** and you would face legal consequence.

- The course strictly prohibits plagiarizing from any source such as another student or online material. Any such activity will be reported to the office of Academic Integrity for dishonesty.

This lab contains materials borrowed from several other lab assignments. I would specially mention Dr. Andrew Kalafut of Grand Valley State University. While the goals of the lab are similar, these instructions are completely rewritten.