CS 3923/6813 - Information, Security and Privacy
Assignment 1, part 2: Password Cracking
DUE: Tue, 19 Feb 2019 @ 6:00PM

This assignment is based on an assignment from Lok Kwong Yan to introduce the students to the concept of password cracking. Now unlike other assignments that the student might be used to, this assignment is designed to make the student search for the solution themselves. As you will see below, I do not tell you which tool you should use or what you need to do. I just give you the problem and then it is up to you to solve it.

The main reason for this is because Computer Security changes extremely fast and it is up to us to make sure that we are comfortable with learning new concepts and ideas on our own as well as try some new found tools or techniques. Note that while I want you to do this yourself, it doesn't mean that I won't answer questions. So don't forget to ask questions.
For this assignment, the student is asked to (READ THE WHOLE ASSIGNMENT FIRST)

1. Download the formspring hash file from. http://isis.poly.edu/~egavas/cs6813/assign2/formspring.tgz
2. Download the first 2M hashes from the Ashley Madison hack from http://isis.poly.edu/~egavas/cs6813/assign2/AM2M.dump.gz. You will have to extract the individual fields from the dump. The dump that I am giving you is in the following format: (id#,username,password_hash,login_key,#). If you only want the password hashes then retrieve http://isis.poly.edu/~egavas/cs6813/assign2/hashes.128.tgz.
3. Do some digging (e.g. search online) to gather some information about the formspring hash file as well as the Ashley Madison hash file. Information such as how the passwords were hashed (e.g. the hashing algorithm) is particularly important. (You might also want to look up information on "crypt" and "shadow" - crypt is the tool used to generate the hash entries for the "shadow" file.
4. Do some digging to find a particular password cracking tool – MAKE SURE IT IS A LEGITIMATE TOOL!!!! DON'T JUST INSTALL ANYTHING. Make sure the tool supports dictionary mode and brute force modes – you will need them later. Also, I suggest that you use the SEED VM for your assignments (http://www.cis.syr.edu/~wedu/seed/lab_env.html). Alternatively KALI Linux (https://www.kali.org/) also contains some password cracking utilities, or AWS can serve as an alternative for a few dollars (and also provides optional GPU support). You have many options.

5. Once you have figured out how to parse the password files, use the tool that you have found to try and crack as many passwords as you can. First try the dictionary mode – this will require you to use find a dictionary file (hint: there are some on github that can be quite helpful). Second, try the dictionary mode again using a different dictionary (I suggest one based on previously cracked passwords.) Third, try the brute force mode to recover as many passwords as possible.
6. If you didn't get any passwords using the dictionary attack, or a very few number of passwords that is fine. You should have gotten some passwords using brute force though. This might take hours – but should only take minutes if you take a smarter approach – but you should get some passwords. You cannot submit a solution without having cracked some passwords.
7. For the formspring list, you should retrieve enough passwords so that you can see a particular pattern which makes up the "salt". HINT: This pattern is most obvious if the "passwords" that you cracked contains both numbers and letters.
8. OPTIONAL: Once you have gathered enough passwords, determine what the Formspring pattern/salt is. Good password crackers will give you the ability to use regular expressions to define patterns to transform dictionaries words. Try to apply the salt rule to the dictionary files you used and see if you crack any more passwords. For example, you might be able to specify something like try all passwords where the first two characters are lowercase letters while the next 6 are only upper case. Another example is for each word in a dictionary, prepend a number to the word to create a new one. This effectively gives you 11x the number of passwords (the dictionary list, the list with 0 prepended, 1 prepended …). Apply the pattern to the larger dictionary that you found and see how many more passwords you were able to crack. A good pattern should result in numerous passwords since people tend to reuse simple passwords (see http://splashdata.com/press/worst-passwords-of-2014.htm).
9. OPTIONAL: Apparently Ashley Madison didn't only use bcrypt (which is great!) to hash the passwords, it also used MD5(username || "::" || password) where || is string concatenation to generate the login_key. Since MD5 is much faster than bcrypt, some of the passwords are much easier to crack than others. See http://arstechnica.com/security/2015/09/once-seen-as-bulletproof-11-million-ashley-madison-passwords-already-cracked/. Convince yourself that this is true by either running some experiments or adjusting the password cracking scheme to account for this particular weakness.

Once you have finished all of the above please submit a PDF report that details the following:

1. Describe the results from step 3 above. In the minimum, you need to describe the hashing algorithms used and the salting technique used. Note that YOU CAN'T just stay that someone said so – you need to explain why YOU believe that what you found is true. For example, you can't just say "According to Wikipedia, formspring used 3DES to encrypt their passwords." That answer only shows me that you can do some searching, but not that you actually understand all of the different concepts. A better answer will look more like "According to Wikipedia, formspring used 3DES to encrypt their passwords. I believe this is true because the password hashes were separated into 64bit chunks and also because I tried a cracker that supports 3DES and the results returned a lot of plausible passwords."
2. Describe the tool that you found and also describe the two different modes – dictionary and brute force. What are the advantages and disadvantages to the different modes?
3. Describe how many passwords you were able to crack using the dictionary attack, the dictionary attack with a separate dictionary and then the brute force attack. Explain why, in your words, it is a good idea to use a list of previously cracked passwords as a dictionary. What did you observe about the cracked passwords? Finally, include screenshots of the different commands that you used, as well as screenshots of the resulting outputs/passwords (doesn't have to be comprehensive – just some output is fine). I just need to know that you were able to crack both passwords dumps.
    a. Did you notice any difference in speed (e.g. passwords cracked per second) between cracking the different passwords? Explain why or why not.
4. Based on your experience, describe (briefly – you can use a bulleted list for this one) the steps you would follow to try to crack passwords in the future. Basically, if you are given a hash file (one with just a bunch of hex values) for company X, what would you do to try to recover as many passwords as possible.
5. Finally, if you were in charge of creating a password scheme, describe at a high level what algorithms and storage methods you would use. This is an open-ended question, so provide as many details as needed to justify your response.