COMP 8005
Computer Systems Technology
April 13th, 2023
Network Security

Project: Distributed Password Cracker

Due: Due: Tuesday, April 11th, 2023, at 1:29 PM.

This is a pairs assignment.

Objective

Create a distributed password cracker for Fedora 36 or 37.

Tasks

- You must write a distributed password cracker to determine user passwords in the /etc/shadow file.
- Use assignment #2 as a starting point
- The program must take the /etc/shadow file and a list of users to determine the passwords as command line arguments.
- The write-up must focus on the performance increase from the assignment 1 single-threaded version.
    - What if you have 1 machine?
    - What if you have 2 machines?
    - What if you have 3 machines?
    - Etc..
- At the very least, you must provide a graph of the different scenarios. Also, what was your expectation vs reality? What are the reasons for the difference between expectation and reality?

Constraints

- May be written in any programming language you like.
- Must run on Fedora Linux.
- Must run like
    - <program invocation> -f /etc/shadow -t # foo bar -p 8005
        - This will determine the passwords for the user foo and the user bar and run the server on port 8005
        - There will be some number of machines that will connect to the server to participate in the password cracking

        - You can add additional arguments as needed for your implementation
- The output of the program must contain:
    - The name of the hashing algorithm used for the user in the /etc/shadow file
    - The password
    - The amount of tries it took to find the password
    - The total time it took to find the password
- You can limit the program by the number of tries or the amount of time if you like (as parameters to the program)
- How you manage the client and server programs is up to you
- You can use any libraries or functions for the hashing part of the program. However, you must write all the other parts yourself
- **You may not use any tool or library that implements anything beyond the hashing algorithm and a threading library if you choose to use one without explicit permission from the instructor.**

Submission Requirements

Use the following directory structure (omit directories that are not needed):

Directory	Purpose
source	Any source code files
report	Report files in .pdf format
video	Video(s) demonstration of your working project
pcap	Any relevant packet captures

Notes

- Follow the appropriate report format (samples).
- The demo video should cover each one of your test cases. In other words, it will be similar to an actual lab demo, except you will prepare a video of each test instead of me observing each test.
- During the test, **if necessary**, you will capture network traffic on any relevant machines and submit the pcap files as specified above.
- Please set the appropriate packet capture filter to limit the size and scope of the data collected to be what is necessary.

Format

You must hand in a pax.Z file to the assignment submission folder on Learning Hub (https://learn.bcit.ca).

You can hand in as many versions as you like. The last submission, based on the timestamp, will be the one to be marked. **Replace # with the actual version number of your assignment (e.g. 1, 2, or 3).**

```
pax -w source/ report/ video/ pcap/ -f project-v#.pax
compress -f project-v#.pax
```

Hand in the resulting project-v#.pax.Z file.

Note: failure to follow the submission requirements may result in a loss of marks, up to 100%.

Only one person in the project has to hand in the file.

Evaluation

Grading:

Item	**%**
Client	10%
Server	30%
Password	10%
Design	20%
Testing	20%
Report	10%

Notes