Concurrency and Parallelism

Degree in Computer Science 2022

Lab 2 – MD5

In many system that need to authenticate using passwords, they are transformed using a hash function. To check if a password is correct, we apply the hash function and compare with the stored hash. If there is a security breach, the passwords are not exposed because the hash function is not easily invertible.

If we restrict the number of possible passwords we can actually break the hash using brute force. The break_md5 program uses the hash to recover passwords with the following restrictions:

- Passwords have 6 characters.
- Characters can only be lowercase letters.

Because there are 26 letters, these restrictions allow for 26^6 different passwords. In break_md5 there are two functions that map numbers from 0 to $26^6 - 1$ to passwords and viceversa. To break a hash, the program iterates from 0 to $26^6 - 1$, generates the corresponding password, calculates its md5 hash and compares it with the hash we want to break.

Given a password, we can compute the md5 hash using md5sum:

```
$ echo -n "passwd" | md5sum
76a2173be6393254e72ffa4d6df1030a  -
```

And we can break the hash using the break_md5 program:

```
$ ./break_md5 76a2173be6393254e72ffa4d6df1030a
76a2173be6393254e72ffa4d6df1030a: passwd
```

Check that you have the development headers for the openssl library installed. (e.g. on debian and ubuntu you should install the libssl-dev package)

Modify the program so that:

Exercise 1 (**Add a progress bar**) Right now we have no way of checking the progress of the program. Add a new thread that prints a progress bar. The thread that is trying to break the hash should communicate with the new one to provide information about its progress. Do not use an active wait.

In order to print the progress bar you can use the special code \r to return the cursor to the beginning of the currect line. That way you can overwrite the progress bar as the state changes.

Exercise 2 (**Add an estimation on the number of passwords checked each second**) Print the number of passwords checked per second next to the progress bar. That total should be updated live, so if the system load increases, and there is less computing power available to check passwords, the number should update on the progress bar.

Exercise 3 (**Make the program multithreaded**) Right now the program makes the brute force attack using just one thread. We could start several and divide the possible passwords between them. If one of them finds the correct password you should stop the others.

Do not divide the passwords that each thread will check when you create them, but rather make them share a counter with the first unchecked password number. This variable should be checked by the threads to get the next password number to check. In order to reduce contention

on the counter the threads should increase the counter by more than 1, and then check all the numbers in that segment all at once (e.g. if the counter is currently 500, a thread could increase the counter to 600, and then test the passwords from 500 to 599 without checking the counter).

Check that the number of passwords checked per second has increased with this new version.

Exercise 4 (**Break several passwords at once**) Using brute force means that we have to generate the hash code and check every possible password. If we wanted to break several passwords it would be faster to check all of them at the same time.

Modify the program so that it receives several hashes on the command line:

```
$ ./break_md5 76a2173be6393254e72ffa4d6df1030a 35bc8cec895861697a0243c1304c7789
```

If a thread breaks a hash code, all the rest should stop checking for that one.

Submission

Assignments are due on March 6. Register for the assignment in github classroom at https://classroom.github.com/a/NROvPR1t. When you register github will create a repository for you with the starting code for the assignment. Push your solutions to that repository. Please fill in your name and login in the authors file.

We will review your submissions during the following week (march 7).