ICS XXX: [Insert Course Title] Spring 2022

Problem Set X: Introduction to Parallelization

Prof. FirstName LastName *Due: Friday, Month XX, 2022 at 11:55 pm*

You may discuss the problems with your classmates, however **you must write up the solutions on your own** and **list the names** of every person with whom you discussed each problem.

1 Overview

After a long break, students easily forget how to code or use simple Linux commands when the new semester begins. Therefore, this assignment is catered to help students refresh their preferred coding language, run and compile the program using the command line, and introduce the topic of parallelization.

Parallelization refers to the process of taking a serial code that runs on a single CPU and spreading the work across multiple CPUs. This greatly speeds up runtime on programs that run longer than a constant factor $O(c)$.

The goal of this assignment isn't to pass but rather encourage students to understand that lazy code isn't practical and optimization is important when working with large data sets.

Students are challenged to write a program that will finish the computations necessary within 1 hour.

2 Dictionary Brute Force Attack (100 pts)

2.1 The Scenario

Suppose a company has hired you as a forensic analyst to attempt a penetration test on their network. You discover that the company's intrusion prevention system is outdated and you can observe decrypted network traffic. You quickly scan through the network and find out how the company generates master passphrases' to access their software.

The master passphrase is generated using a key and a salt. The key is created by hashing a word from a dictionary. Similarly, the salt is created by hashing a word from a dictionary. The words are chosen at random from a dictionary of 466,550 words.

The sum of the key and the salt creates a new passphrase that will allow access to their software.

$$master\_passphrase = key + salt \tag{1}$$

Most importantly, a new master passphrase is generated every hour.

2.2 The Hashing Algorithms

Both algorithms will return a variable with type unsigned. The easiest way to calculate the sum is to add the unsigned values. Keys use the JENKINS(key, size) hashing algorithm, and salts use the not so great LOSE-LOSE(s) hashing algorithm. The algorithms are provided below.

Algorithm 1

```
function JENKINS_ONE_AT_A_TIME(key, size)                ▷ Creates a hashed key
    uint32_t $hash, i$
    for $hash = i = 0; i \rightarrow len; i++$
        $hash \mathrel{+}= key[i]$
        $hash \mathrel{+}= hash << 10$
        $hash \mathrel{\hat{}}= hash >> 6$
    $hash \mathrel{+}= hash << 3$
    $hash \mathrel{\hat{}}= hash >> 11$
    $hash \mathrel{+}= hash << 15$
    return $hash$
```

Algorithm 2

```
function LOSE-LOSE(s)                                    ▷ Creates a hashed salt
    uint32_t $hash$
    for $hash = 0, \cdot s \mathrel{!}= '\backslash 0', s++$
        $hash \mathrel{+}= \cdot s + (31 \cdot hash)$
    return $hash$
```

2.3 The Assignment (100 pts)

Design an program that will successfully find the key, salt, and the master passphrase that is generated at a specific hour before the time is up. [*Hint: How can computations can your machine do on a single thread?*]

Here is a GitHub repository to get you started.

3 Input/Output Formats

To **compile**:

```
$ make <programName>
```

To run with **specific** indexes:

```
$ (time ./programName 1000) >> a.out 2>&1
```

To run with **random** indexes:

```
$ (time ./programName) >> a.out 2>&1
```

References

[1] Jenkins, Bob (November 3, 2013). "A hash function for hash Table lookup". Retrieved December 5, 2021.

[2] Brian Kernighan, Dennis Ritchie (1978 First Edition). "The C Programming Language". Retrieved December 5, 2021.