Michael Lee

Programming Assignment 1

1. For this project I chose to use mostly cryptographic hashes for simplicity: I used sha, md5, and murmur3. The reason I used these hashes is, with the exception of murmur, that they don’t require additional libraries they’re all included within the hashlib module of python. I used the mmh3 library to get the murmur hash. I tried to get non-cryptographic hashes because for a bloom filter they are much faster, but ran into issues with additional libraries. The output range of these hash functions vary a lot, but I truncated them to 10 million with modulos. The size of the bloom filter thusly is 10 million sized bit bucket, and I kept this consistent for both the 3 hash and 5 hash bloom filter
2. To check one password the time required is essentially the amount of time it takes for the program to perform 3 or 5 hash functions depending on the bloom filter used. Because I didn’t change the size of the bloom filter with 5 hashes, the 5 hash filter will have more false positives than the 3 hash filter and be slower depending on how many insertions there are. In our case there weren’t that many insertions (60k), and so the 5 hash filter will have less false positives.
3. False Positive Rate: (1-e^-kn/m)^k n=623518, m=10000000 k=3/5
   1. k = 3: .005
   2. k = 5: .0014

There is no chance of a False negative in our or any bloom filter, if the item is not in the set it won’t report it being in the set. Hashes will always hash to the same value and if it’s always inserted correctly then we will know that a bucket is empty.

1. To reduce the rate of false positives we can increase the size of the bloom filter (increase m) decrease the insertions (n) or increase the number of hashes at the cost of time.