CS161–Fall 2014 — HW4 Write-Up

Erik Bartlett, `cs161-en`

October 22, 2014

My implementation of gentable is in gentable.c. I first calculate the maximum size of my output file in number of password/final chain entry pairs that can be written within the bounds specified by $3 * 16 * 2^s$ bytes. I then go through each possible password, starting at 0 and compute its hash-reduce chain for $2^(n - s)$ cycles. The pair is only written to the rainbow table if it does not collide with itself before 2^{n-s-1} hash-reduce cycles. I used a bitmap of the possible passwords to keep track of the passwords I had seen in any given chain - and only worried about collisions in the local chain as opposed to the number of global collisions. Once I have written as many pairs as is allowed by the space constraint gentable.c is finished.

Crack's implementation uses many of the same functions as gentable.c. I again start by calculating similar constants to those of gentable.c (output table size, etc) and then do a loop to find a matching password to the hash. In this loop I call a method "match" that takes the open rainbow table file and reads through every hash-chain value with the current reduced password until it finds a match, and then it copies that chain's initial password into a given pointer. Then, one hash chain at a time, the password's hash is compared to the input hash to see if there is a match. If there is a match it is returned; if the password is hashed to a point that it is past the point of matching in the chain the password is hashed and the loop is started over, looking for a new match. This goes on until it has matched with a password and is able to find a hash within that hash-chain that is equivalent to the hash given.

As far as the relationship between s and the number of AES operations expected, the number of entries in the table grows proportionally with 2^s - meaning that larger s leads to fewer amounts of AES operations to get the same number of comnparisions to matches. But the number of AES operations needed on average to find the correct hash in the chain also grows exponentially in s, as that is how the length of the chain grows. Therefore the number of AES operations needed on average must grow at a rate of 2^s.

A 20-bit password with hash 0xae60abdcb19d5f962a891044129d56d4:
A 24-bit password with hash 0xeb94f00c506705017ce61273667a0952:
A 28-bit password with hash 0xa2cf3f9d2e3000c5addea2d613acfda8: