Anthony Constantino
1/26/17

Hash Attack Report

Introduction

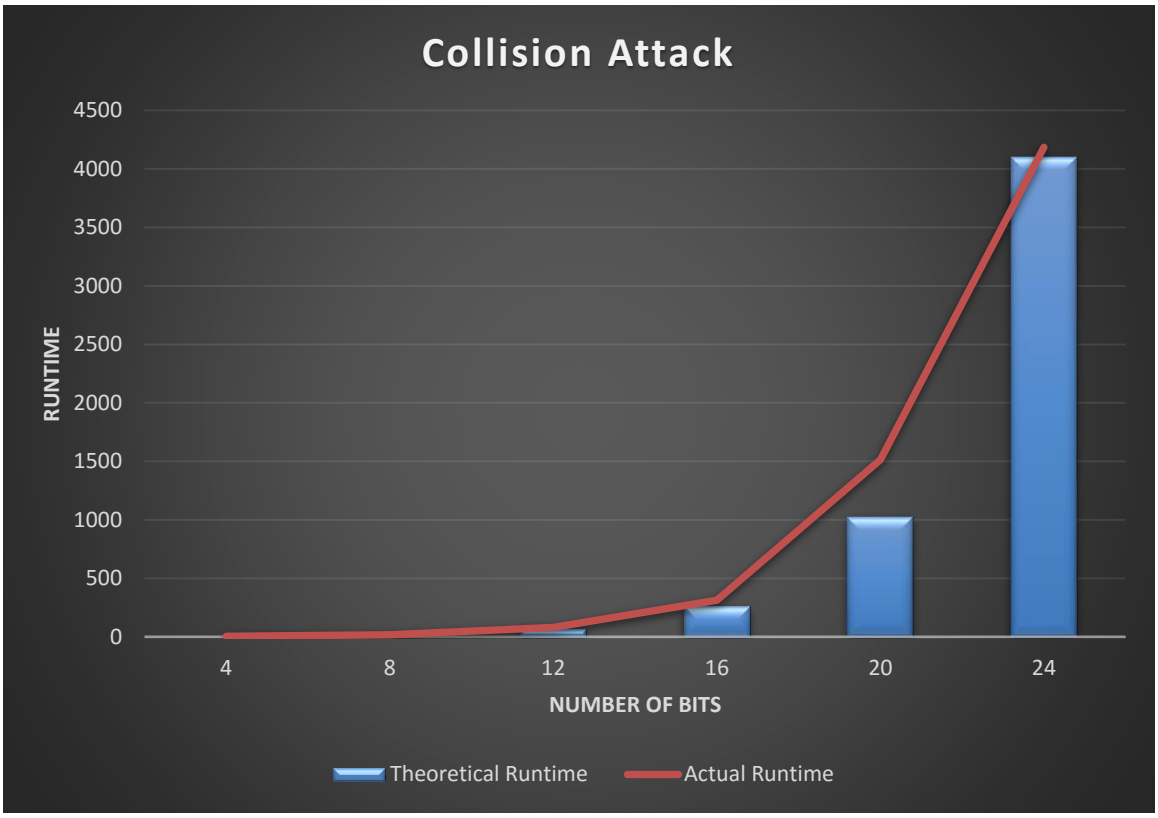
Many times, secret messages such as passwords are encrypted using hashes to hide the characters in them. Two commonly known methods hackers have used to get access to messages/passwords are called collision attacks and pre-image attacks. The first, collision attacks, is the process of generating many random sets of characters and then hashing them in attempt to get a set to hash to the same value as the password. On the other hand, a pre-image attack is one where the set of characters is hashed, and then many random sets of characters are generated and hashed in an attempt to find other sets that hash to the same value as the original set. Exploring these methods will show that they are not as effective as some would think.

Experiment

To test these methods out I have written a program that will simulate these types of attack looking for hashed collisions like the examples stated previously. To do this the program will be generating sets of characters that will be at least 8 characters long to simulate an actual password as most places will require that passwords are at least 8 characters long and have a number/symbol. These messages can be much larger, but to keep things simple and run times short, we will cap the message to 20 characters and will only be using subsets of those generated passwords to allow the program to finish in reasonable time while still being able to show how long it would take to get one of these attacks to work.

Collision Attack

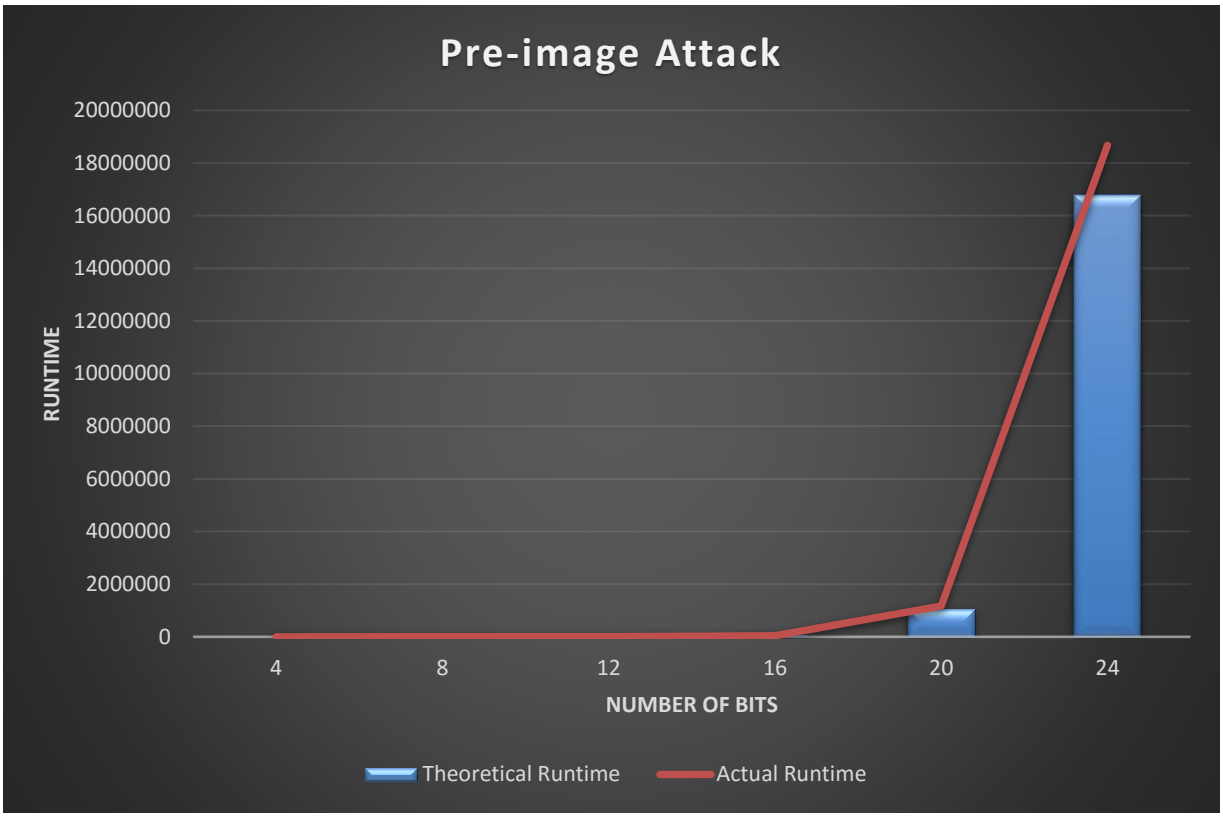
For the collision attack the following will be the algorithm and analysis of the results against the theoretical time of the attack to work. For the algorithm, first we will set up a key-value pair, whose key will be the hashed value and the value will be the set of characters used. Next until there is a collision the program generates random sequences of characters, of sizes specified earlier, creating a hash of the sequence and checking to see if the key is in the key-value pair. Once there is a duplicate hash key the program stops and prints how many attempts were needed to encounter a collision. Record the results and move onto the next size up.



Now let's take a look are the results. Here is a chart showing the actual results and how they stack up to the Theoretical. The theoretical time for this kind of collision to take place is 2^(n/2) where n is the number of bits being hashed. As we can see the actual results were really close to what it to be expected. As the number of bits increases the time increases exponentially, and the time for just 24 bits has started getting pretty large. When translating these, each 4 bits is the equivalent of 1 character. Meaning that if you had even a password of 8 or more characters which is 32 or more bits the time it would take to find a collision would be very large.

Pre-Image Attack

For the collision attack the following will be the algorithm and analysis of the results against the theoretical time of the attack to work. For this algorithm to work, first will generate a random string to represent our password. Then we will hash it and remember that hash. Now we will generate random sequences of characters and hash them in an attempt to get the hash of a random sequence of characters to match the original hash of the generated password. After we find a duplicate hash of the password we will stop and print out how long it took to reach the duplicate hash. Record the results and move to the next size up.



Now let's take a look at the pre-image attack results in the same way we looked at the collision attack. The theoretical time for this kind of collision to take place is 2^(n) where n is the number of bits being hashed. Again, we can see the actual results were really close to what it to be expected, which in this case it is also an exponential amount of time.

Analysis

As the test results show, for either of these attacks to work it would take a great deal of time and computing power. This is most likely why most passwords today are required to be at least 8 characters long and have a number/symbol in them. The variety of characters would increase the set of characters possible in the password, and the length would increase the time needed to go through all possible combinations of these characters. Unfortunately, as time progress machines are getting more powerful and sophisticated to the point where a password containing 8 characters with 1 number/symbol will be too simple for machines to crack, so I would recommend thinking of one that is better than the minimum requirements. Protect your password!