Farid Rajabinia - fr2md

Based on a block structure of the MD5 and the algorithm of Wand and Yu, it is possible to create a vulnerability in MD5 hash. The length of the file should be a multiple of 64 bytes. Then we divide it into those 64 bytes blocks and we hash them using a sequence of 16-byte states. MD5 hash has avalanche effect meaning that the hash of the next block is dependent on the hash of its previous block. The hash of the first block is fixed and called initialization vector. Evilize creates that i vector for our purpose. We start with a c file with two tiers – two main functions. We try to create two binary files with the same hash from this c file.

MD5col is responsible for brute forcing hashes of two pair of blocks and then make sure that they have the same hash values to create two separate files with the same hash. In order to create two different strings with the same hash, we need to make sure to find pairs of blocks in each string that are either the same blocks or different but with the same hash values. We can do it by brute forcing each pair of 64-byte blocks from each of the strings, and check if the blocks are the same with the difference of 128 bytes somewhere in the middle of the string. Then we hash the string and we should get the same hash for both of them. For example, in the following blocks if the pairs in red have the same hash value ($f(f(s_i, M_i), M_{i+1}) = f(f(s_i, N_i), N_{i+1})$) and the rest of the strings are the same then we get the same hash value for both of the strings. f() is a function that calculates the hash.

$M_0, M_1, ..., M_{i-1}, M_i, M_{i+1}, M_{i+2}, ..., M_n,$

$M_0, M_1, ..., M_{i-1}, N_i, N_{i+1}, M_{i+2}, ..., M_n.$

We need to start this search of same blocks or same hashed block based on a fixed initial point that is i vector. Creating a first program is dummy, we just create hash of the first blocks and call it the first program. For the second program, we are searching for the pair of blocks that are identical to the blocks of the first program with the difference of 128 bytes in the middle of the blocks containing a different pair but with the same hash value. The brute forcing of each pair of 64-byte blocks - by trying to change the values in the pair while making sure the hash of the altered pairs are still the same - took 5 hours of computing on my laptop (I7, 2.3 GHz on a VM Linux).