Faris Zahrah
Practical Cryptology
Meet In The Middle
November 27, 2018

A meet in the middle attack on double DES with the reduced key size is performed by attacking the first des encryption from the plaintext and attacking the second des encryption from the final cipher text. The algorithm looks for key1 and key2 such that when key1 encrypts the plain text and key2 decrypts the cipher text, they obtain the same intermediary block.

Because of the key size and block size of DES, collisions are expected in the attack. This conclusion comes from the analysis of MITM and the equation relating key size and block size. To reduce the number of collisions, one may increase the block size such that less different keys map to the same block.

Attached implementation details

In the attached program the strategy I employed was to create a dictionary of all potential key1 values and their encrypted block with the given plain text. This is the list I used to compare with blocks that I obtain from iterating through all potential key2 values and retrieving the decrypted ciphertext. Based on the relative key and block sizes, I expected collisions. So I made a second dictionary of key1 values and their correlated encrypted blocks, that was only inserted into if the block was already inserted into the initial list. I also made a counter to ensure that there were no collisions in the collisionBlocks list. If there were any collisions, then a third list must be created to ensure that all potential keys are checked.

The initial list to compare block values against is O(n) in terms of space and time complexity. The second part of comparing blocks with key2 values is O(n) in terms of time complexity and O(1) in terms of space complexity if you can assure zero or roughly 1 collision.

The time to attack this cipher is the amount of time to encrypt/decrypt DES 2*2^20. Because we have to compute all potential intermediate block values with all key1 values. This is 2^20. And we then do the similar amount of computations in the reverse direction and check for intermediate block matchings. So this is 2^20 + 2^20. Which is 2^21.

The effective key length is 21 bits. This is because the key length with key1 only is 20 bits and the key length of key2 is only 20 bits and solving for both does not require squaring the operations but rather adding the sum of operations.